

Task-Based Prescriptive Trees for Two-Stage Linear Decision-Making Problems

Rafaela Ribeiro

Pontifical Catholic University of Rio de Janeiro, Department of Industrial Engineering
Rua Marquês de São de Vicente, 225 – Gávea, Rio de Janeiro, RJ – Brazil – 22541-900
rafaelamaribeiro@aluno.puc-rio.br

Bruno Fanzeres¹

Pontifical Catholic University of Rio de Janeiro, Department of Industrial Engineering
Rua Marquês de São de Vicente, 225 – Gávea, Rio de Janeiro, RJ – Brazil – 22541-900
bruno.santos@puc-rio.br

RESUMO

A maioria dos problemas de tomada de decisão pode ser classificada como problemas estocásticos de dois estágios. O *framework* clássico segue um processo sequencial de duas etapas. Primeiramente, estima-se uma distribuição preditiva baseada em métodos de *machine-/statistical-learning* (M/SL), e depois prescreve-se uma decisão resolvendo o problema estocástico de dois estágios considerando a distribuição estimada. Tipicamente, métodos M/SL focam apenas em minimizar o erro de predição de parâmetros incertos, desconsiderando o impacto no problema de decisão. Entretanto, o maior interesse é obter soluções com menores erros de decisão ao invés de decisões dirigidas por menores erros de predição. Neste trabalho, discutimos um novo *framework* em que a função de treinamento M/SL também considera o problema de decisão. Focamos em árvores de decisão, apresentando o modelo MILP exato e uma heurística de partição recursiva para este. Finalmente, analisamos um conjunto de experimentos ilustrando a efetividade do método proposto, comparando-o com *frameworks* clássicos.

PALAVRAS CHAVE. Análise Prescritiva; *Decision Focused Tree*; Otimização.

Tópicos: IC – Inteligência Computacional, BDA – Big Data e Analytics, PM – Programação Matemática

ABSTRACT

Most decision-making problems are framed as two-stage stochastic programs. The standard framework follows a sequential two-step process: a predictive distribution is firstly estimated, leveraging on machine-/statistical-learning (M/SL) methods; then, a decision is prescribed by solving the two-stage stochastic program using the estimated distribution. M/SL methods typically focus only on minimizing the uncertain parameters prediction error, leaving out the impact on the decision problem. However, practitioners' main interest is to obtain prescriptions under uncertainty with minimum decision error rather than decisions oriented by least-error predictions. In this work, we propose a new framework where the M/SL training function also accounts for the downstream optimization problem. We focus on decision trees, presenting an exact non-convex mathematical programming-based formulation and also a recursive-partitioning heuristic strategy to identify good and feasible solutions to the non-convex problem. Finally, we analyze a set of experiments illustrating the effectiveness of the proposed methodology, benchmarking against standard frameworks.

KEYWORDS. Prescriptive Analytics; Decision-Focused Tree; Optimization.

Paper topics: CI – Computational Intelligence, BDA – Big Data and Analytics, MP – Mathematical Programming

¹This study was partially supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and by Rio Paraná Energia S.A. Rafaela Ribeiro was partially supported by CNPq, project 88887.877556/2023-00, and FAPERJ, project E-26/202.291/2024. Bruno Fanzeres gratefully acknowledges the financial support from CNPq, project 309064/2021-0, and FAPERJ, project E-26/202.825/2019.

1. Introduction

In the past decades, the pipeline for devising decisions under uncertainty has traditionally followed an *estimate-then-optimize* framework [Elmachtoub and Grigas, 2022], constructed upon the historical/observed data that directly and implicitly impacts the agent’s goal targets. In this context, following standard statistical theory, firstly, the future dynamic of the uncertain variables is characterized by an (unknown) predictive distribution, whose estimation is based mainly on the observed data of the real stochastic process and a set of side information, usually referred to as *covariates*, which oftentimes help to improve the quality of the predictive distribution estimation by providing some level of contextual drive. Typically, the framework for such predictive distribution estimation aims at minimizing a predictive error metric, such as the mean-squared or mean-absolute error, whose mismatch impact between prediction and actual realization has a symmetric cost [Christoffersen and Diebold, 1997; Lanchier, 2017]. Then, a sample-based surrogate model of the decision problem is constructed and solved using standard or specialized optimization algorithms. The most common approach to constructing such a surrogate decision model is known as the Sample Average Approximation (SAA). Essentially, the SAA approach is built on an approximation of the value-functional (e.g., expected value) by accounting for its sample counterpart, leveraging on strong convergence guarantees between both uncertainty estimation functions and the sufficient finite-sample properties [Shapiro et al., 2009; Parys et al., 2021]. The most common methods for sampling and value-functional approximation are based on Statistical/Machine Learning techniques using, for instance, Monte Carlo or Importance sampling, *k*-nearest neighbors, Random Forests, and Neural Networks [Sen and Singer, 2019; Parys et al., 2021].

However, it has been recognized in the past years by both the scientific community and practitioners that a prescription under uncertainty should be based on optimal solutions with minimum decision errors rather than the ones oriented by a least-error predictive distribution. The rationale for this is that decision errors are intrinsically asymmetric to the decision-maker, but predictive distributions have been recurrently estimated using symmetric loss functions. Therefore, a misalignment between the main objectives of a symmetric-based estimation for the predictive distribution and its use in the decision-making process is noticed, encouraging the development of novel methodologies that jointly meet the utmost objective of a prescription with minimum decision error. In the past years, different approaches have appeared in the scientific literature to enhance decision-making under uncertainty processes in the aforementioned prescriptive-oriented direction from multiple angles. Elmachtoub and Grigas [2022] designed a framework named “*smart predict-then-optimize*” which directly leverages the downstream optimization problem structure for designing better prediction models whose loss function is built to measure the decision error induced by a prediction. Stratigakos et al. [2022] proposed a *Prescriptive Tree* that integrates forecasting and optimization into the training procedure aiming at minimizing overall decision costs with an application in renewable energy trading problems. Furthermore, Estes and Richard [2023] made use of linear regression models to make point predictions for uncertain parameters of two-stage linear decision problems, accounting for not only regression errors but also the optimization problems themselves. Kallus and Mao [2023] studied new machine learning methods based on an ensemble of decision trees that choose their structure, focusing on optimizing the decision quality instead of the prediction accuracy.

Notwithstanding the relevance of the aforementioned literature, none of them improved the standard estimate-then-optimize framework toward a predictive- and prescriptive-oriented uncertainty distribution estimation for an optimal prescription under uncertainty for general decision problems representable as two-stage stochastic programs. In this work, we focus on closing this gap by extending the decision tree framework. Roughly speaking, standard *Decision Trees* is a Machine Learning method that aims at finding partitions in the uncertainty support set (referred to as the leaves of the tree) such that a predictive-oriented loss function is jointly minimized in each partition/leaves [Bertsimas and Dunn, 2019]. The reasoning behind the method is that each leaf contains a similar uncertainty structure and dynamics and is intrinsically different from the other leaves. Thus, only “similar data” is considered in the decision process, enhancing the quality of the predictive distribution characterization for prescription purposes [Bertsimas and Dunn, 2019; Bertsimas and Kallus, 2020]. We formalize the predictive- and prescriptive-oriented uncertainty

distribution estimation for an optimal prescription under uncertainty for general decision problems representable as two-stage stochastic programs using decision trees as a non-convex mathematical program. Furthermore, recognizing the difficulty in handling the proposed non-convex mathematical program with standard optimization algorithms or off-the-shelf solvers, we design a recursive-partitioning heuristic strategy following the ideas discussed by Breiman et al. [1984] and Bertsimas and Dunn [2019]. Essentially, the heuristic strategy consists of sorting all historical data points in ascending order, and the potential partition is a mid-point between consecutive covariate observations, generating two partitions. After a split is found, the procedure is recursively called in each of the two generated partitions and proceeds up to a stopping criterion (e.g., maximum depth or minimum number of observations in the leaves) is met.

To validate the prescription capability of the proposed decision-making framework, a collection of numerical experiments is conducted using three typical two-stage decision problems. The first one is referred to as the *Two-Stage Shipment Planning Problem* (TS-SPP) [Bertsimas and Dunn, 2019], the second one the *Stochastic Shortest-Path Problem* (S-SPP) [Elmachtoub and Grigas, 2022], and the third is based on the *Two-Stage Farmer's Problem* (TS-FP) [Birge and Louveaux, 1997]. We choose these problems since they possess a particular nature of the uncertain impact on them. More specifically, from a mathematical perspective, the uncertainty impacts the TS-SPP on the right-hand side, the S-SPP in the objective function, and the TS-FP in the so-called technology matrix. The proposed methodology is compared to the following four typical benchmarks found in the technical literature for predictive distribution estimation: k -Nearest Neighbors (k NN), Random Forests, Nadaraya-Watson Kernel Regression, and Neural Networks. The comparison will be evaluated considering a metric referred to as the Coefficient of Prescriptiveness (P), proposed by Bertsimas and Kallus [2020], that aims at comparing the prescriptive capability of the methods with the one using all historical data in an SAA framework, regularized by a decision under Perfect Information. The experiments indicate that, for all test problems, the proposed methodology consistently performs superiorly to the benchmarks and also to the purely data-driven SAA.

2. Task-Based Prescriptive Trees for Two-Stage Linear Decision-Making Problems

In this work, we focus on constructing an efficient methodology for learning optimal task-based prescriptive trees for decision-making problems representable by the following two-stage linear mathematical program [Birge and Louveaux, 1997; Shapiro et al., 2009]:

$$\min_{\mathbf{z} \in \mathcal{Z}} \left\{ \mathbb{E}_{F_{\tilde{\xi}}} [\phi(\mathbf{z}, \tilde{\xi})] \right\}, \quad \text{with } \phi(\mathbf{z}, \tilde{\xi}) = \mathbf{c}^\top \mathbf{z} + \min_{\mathbf{y} \geq 0} \left\{ \mathbf{q}^\top(\tilde{\xi}) \mathbf{y} \mid \mathbf{W} \mathbf{y} \geq \mathbf{d}(\tilde{\xi}) - \mathbf{T}(\tilde{\xi}) \mathbf{z} \right\}. \quad (1)$$

In (1), $\mathcal{Z} \triangleq \{\mathbf{z} \in \mathbb{R}^{n_z} \mid \mathbf{A} \mathbf{z} \geq \mathbf{b}\}$ is a nonempty polyhedron set and $\phi : \mathcal{Z} \times \Xi \rightarrow \mathbb{R}$ is a *convex Carathéodory* function², with $\Xi \subseteq \mathbb{R}^{m_\xi}$ denoting the support set of the random vector $\tilde{\xi}$ under the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. As usual, $F_{\tilde{\xi}} \triangleq \mathbb{P} \circ \tilde{\xi}^{-1} : \mathbb{R}^{m_\xi} \rightarrow [0, 1]$ denotes the cumulative probability function of $\tilde{\xi}$ measured by $\mathbb{P}$. We further recognize that the decision-maker acknowledges a set of relevant contextual information $\tilde{\mathbf{X}} \in \times_{i=1}^{m_x} \mathcal{X}_i \subseteq \mathbb{R}^{m_x}$ (hereinafter referred to as *covariates*, following standard nomenclature) that might correlate with the main uncertain vector $\tilde{\xi}$. Furthermore, within this setting, all covariates are revealed with high certainty before making the prescription $\mathbf{z}$. Therefore, following the decision-making structure (1) and assuming an event $\{\tilde{\mathbf{X}} = \mathbf{x}\}$ of the covariates, the problem of interest resumes to

$$\min_{\mathbf{z} \in \mathcal{Z}} \left\{ \mathbb{E}_{F_{(\tilde{\xi} \mid \{\tilde{\mathbf{X}} = \mathbf{x}\})}} [\phi(\mathbf{z}, \tilde{\xi}) \mid \{\tilde{\mathbf{X}} = \mathbf{x}\}] \right\}, \quad (2)$$

with $F_{(\tilde{\xi} \mid \{\tilde{\mathbf{X}} = \mathbf{x}\})}$ denoting the conditional cumulative distribution of $\tilde{\xi}$ on the subspace of events $\{\tilde{\mathbf{X}} = \mathbf{x}\}$.

In the past decades, the standard approach to tackle problem (2) followed an *estimate-then-optimize* framework [Elmachtoub and Grigas, 2022; Kannan et al., 2024], in which the decision-maker possesses a dataset of $|\mathcal{K}|$ pairwise-linked observations of the uncertain parameters $\{(\mathbf{x}_k, \xi_k)\}_{k \in \mathcal{K}}$, and use

²We refer to as *convex Carathéodory* a function $f : \mathcal{Z} \times \Xi \rightarrow \mathbb{R}$ that is convex in $\mathcal{Z}$ for all ξ almost-everywhere and is measurable in the sigma-field of Ξ for all $\mathbf{z} \in \mathcal{Z}$.

it to infer a predictive distribution for the uncertain vector $\hat{F}_{(\tilde{\xi}|\{\tilde{\mathbf{X}}=\mathbf{x}\})}$ conditional to the subspace of events $\{\tilde{\mathbf{X}} = \mathbf{x}\}$, to then devise a solution for the decision problem by solving a surrogate version of (2) built upon $\hat{F}_{(\tilde{\xi}|\{\tilde{\mathbf{X}}=\mathbf{x}\})}$. In this context, for a given hypothesis class $\Psi \triangleq \{\psi : \mathbb{R}^{m_x} \rightarrow \mathbb{R}^{m_\xi}\}$ for predicting the uncertain vector $\tilde{\xi}$, the predictive-distribution inference method usually applied in technical literature leverages the following empirical risk minimization setup

$$\min_{\psi \in \Psi} \left\{ \sum_{k \in \mathcal{K}} \ell(\psi(\mathbf{x}_k), \xi_k) \right\}, \quad (3)$$

with $\ell : \mathbb{R}^{m_\xi} \times \mathbb{R}^{m_\xi} \rightarrow \bar{\mathbb{R}}$ denoting a *prediction-oriented* loss function that measures prediction errors (e.g., the mean squared error $\ell(\psi(\mathbf{x}_k), \xi_k) = \|\psi(\mathbf{x}_k) - \xi_k\|_2^2$), hence not accounting for the impact of the predictive distribution estimation on the downstream optimization problem. However, for many practitioners, their main interest is to obtain near-optimal solutions from the available data with minimum *decision error* rather than a *least-error prediction*. By acknowledging this end, in the next subsection, we present a holistic end-to-end framework based on decision trees with an integrated estimation-and-optimization process to enhance the prescription capability for decision-making problems representable by the two-stage linear mathematical program (2).

2.1. Decision-Tree Training Problem: A Predictive- and Prescriptive-Oriented Approach

The pipeline for decision-making proposed in this work also leverages the empirical risk minimization setup in (3), but accounting for hypothesis classes built upon decision-tree-based ones $\Psi_\theta^{(\text{tree})}$ with an *a-priori* specified set of hyperparameters θ (e.g., maximum depth and minimum leaf length). Roughly speaking, the standard reasoning behind decision trees is to partition the covariate space into L disjoint subsets of $\mathcal{X}$, $(\mathcal{L}_1, \dots, \mathcal{L}_L)$, referred to as “*leaves*” in the technical literature, following a collection of “*splitting rules for partitioning*” characterized by a hierarchical sequence of φ -parameterized disjunctive functionals $h_\varphi : \mathcal{X} \rightarrow \mathbb{R}$, whose predictions in each leaf $l \in \{1, \dots, L\}$ collectively minimize a prediction-oriented loss function (e.g., mean squared error) [Breiman et al., 1984; Bertsimas and Dunn, 2017; James et al., 2023]. Figure 1 illustrates the idea for a partition of the covariate space into three “*leaves*” following two “*splitting rules*” characterized by the functionals $h_{\varphi_1} : \mathcal{X} \rightarrow \mathbb{R}$ and $h_{\varphi_2} : \mathcal{X} \rightarrow \mathbb{R}$.

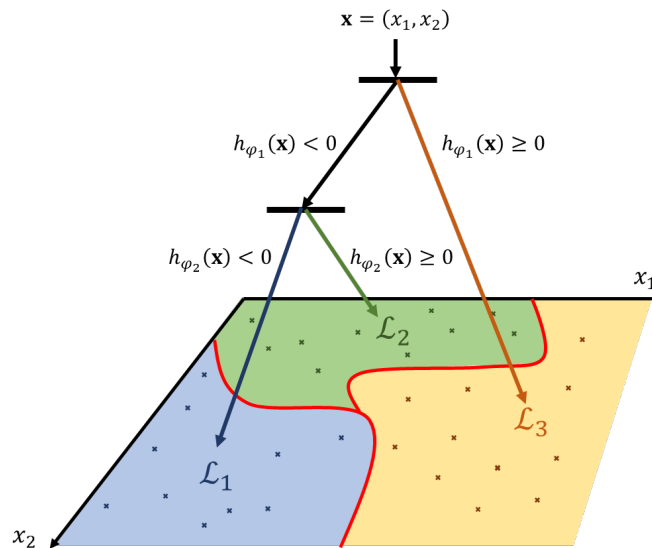


Figure 1: Illustration of the partition of the covariate space into three “*leaves*” following two “*splitting rules*” characterized by the functionals $h_{\varphi_1} : \mathcal{X} \rightarrow \mathbb{R}$ and $h_{\varphi_2} : \mathcal{X} \rightarrow \mathbb{R}$.

A formal representation of this structure is presented in (4).

$$\min_{(\mathcal{L}_1, \dots, \mathcal{L}_L) \in \Psi_{\theta}^{(\text{tree})}} \left\{ \sum_{l=1}^L \left(\min_{\psi_l \in \Psi_l} \left\{ \sum_{k \in \mathcal{K} \mid \mathbf{x}_k \in \mathcal{L}_l} \ell(\psi_l(\mathbf{x}_k), \boldsymbol{\xi}_k) \right\} \right) \right\}. \quad (4)$$

Problem (4) is usually referred to as the decision-tree training problem. In (4), $\Psi_{\theta}^{(\text{tree})}$ restricts the allocation of observations to leaves following the structure of a decision tree, with a prediction-oriented *loss function*. For instance, in the usual context of a mean-squared-error *loss function* with a hypothesis class defined as a constant, covariate-independent prediction value $\Psi_l = \{\psi_l(\mathbf{x}) = \hat{\boldsymbol{\xi}}_l, \forall \mathbf{x} \in \mathcal{L}_l\}$ for all leaves $l \in \{1, \dots, L\}$, following standard statistical inference theory, the solution for the inner minimization problem of (4) is the average of the observations within the leaf $\mathcal{L}_l$ [Casella and Berger, 2002]. On the other hand, in [Bertsimas and Dunn, 2019], the hypothesis class $\Psi_l = \{\psi_l(\mathbf{x}) = \beta_l^\top \mathbf{x}, \forall \mathbf{x} \in \mathcal{L}_l\}$, defined by all linear prediction functions based on the covariates within its leaf $\mathcal{L}_l$, is used. Anyhow, the conditional predictive distribution derived from the decision tree is estimated by focusing on achieving the highest prediction accuracy possible.

In this work, we adapt this general structure to also account for the impact of the predictive distribution on the downstream optimization problem when training the tree. More specifically, let $\mathcal{R} : \mathcal{Z} \times \Xi \rightarrow \mathbb{R}$ be a *task-based* loss function defined as

$$\mathcal{R}(\mathbf{z}, \boldsymbol{\xi}) \triangleq \phi(\mathbf{z}, \boldsymbol{\xi}) - \min_{\hat{\mathbf{z}} \in \mathcal{Z}} \left\{ \phi(\hat{\mathbf{z}}, \boldsymbol{\xi}) \right\}. \quad (5)$$

The *task-oriented* loss function in (5) has a distinct representation than the standard predictive-oriented ones (e.g., mean squared error) since it depends mostly on a prescription rather than solely on the prediction of $\boldsymbol{\xi}$. Structurally, it is a particular composition of the so-called *regret functions* where the “reference action” (second term in (5)) is defined as the optimal prescription under perfect information, thus leading to a measure of divergence/regret over perfect prediction-then-prescription.

Therefore, in this work, we propose to estimate the (conditional) predictive distribution by leveraging, on the one hand, the decision-tree-based structure with, on the other hand, the following predictive- and prescriptive-oriented loss function.

$$\Phi_{\lambda}(\mathbf{z}, \psi, \boldsymbol{\xi}, \mathbf{x}) = \lambda \mathcal{R}(\mathbf{z}, \boldsymbol{\xi}) + (1 - \lambda) \ell(\psi(\mathbf{x}), \boldsymbol{\xi}) \quad (6)$$

In (6), $\lambda \in [0, 1]$ weights the impact of the predictive and prescriptive components in the loss function. More specifically, by setting $\lambda = 0$, the standard prediction-oriented training process is recovered, and as λ increases, the impact of the predictive distribution estimation on the prescriptive capability becomes more emphasized. In this context, the proposed decision tree training problem with the predictive- and prescriptive-oriented loss function (6) can be formally stated as follows:

$$\min_{\substack{\mathcal{L}_1, \dots, \mathcal{L}_L, \\ \mathbf{z}_l^*, \psi_l}} \sum_{l=1}^L \left(\sum_{k \in \mathcal{K} \mid \mathbf{x}_k \in \mathcal{L}_l} \left(\Phi_{\lambda}(\mathbf{z}_l^*, \psi_l, \boldsymbol{\xi}_k, \mathbf{x}_k) \right) \right) \quad (7)$$

subject to:

$$(\mathcal{L}_1, \dots, \mathcal{L}_L) \in \Psi_{\theta}^{(\text{tree})}; \quad (8)$$

$$\psi_l \in \Psi_l, \quad \forall l \in \{1, \dots, L\}; \quad (9)$$

$$\mathbf{z}_l^* \in \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{k \in \mathcal{K} \mid \mathbf{x}_k \in \mathcal{L}_l} \phi(\mathbf{z}, \boldsymbol{\xi}_k) \right\}, \quad \forall l \in \{1, \dots, L\}. \quad (10)$$

Problem (7)–(10) seeks to identify an optimal partition of the dataset (covariate space) into leaves of a decision tree (equation (8)) that minimizes the predictive- and prescriptive-oriented loss function

(7) using as prediction the best-fit function within the set of hypothesis class Ψ_l (equation (9)) and as prescription the solution of the downstream optimization problem (equation (10)), both restricted to the observations in the respective leaf $l \in \{1, \dots, L\}$. Structurally, constraint (8) focuses on specifying the partition of the dataset into the leaves (see Figure 1 for a visual representation), accounting for best predictability fit (9) and prescriptive capability (10) that minimizes the predictive- and prescriptive-oriented loss function (7).

Finally, given an optimal partitioning $(\mathcal{L}_1^*, \dots, \mathcal{L}_L^*)$, as a solution for the training problem (7)–(10), the prescription $\mathbf{z}^*(\mathbf{x}_{\text{new}})$ for a given new covariate data $\mathbf{x}_{\text{new}}$ follows a two-step procedure: (i) identify which leaf $l \in \{1, \dots, L\}$ the new covariate data falls into, hereinafter referred to as $l_{\mathbf{x}_{\text{new}}}$ for expository purposes; then (ii) solve a surrogate version of (2) by replacing the conditional distribution of ξ by the empirical one constructed with the data within $\mathcal{L}_{l_{\mathbf{x}_{\text{new}}}^*}^*$ as in (11).

$$\mathbf{z}^*(\mathbf{x}_{\text{new}}) \in \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \frac{1}{|\{k \in \mathcal{K} \mid \mathbf{x}_k \in \mathcal{L}_{l_{\mathbf{x}_{\text{new}}}^*}^*\}|} \sum_{k \in \mathcal{K} \mid \mathbf{x}_k \in \mathcal{L}_{l_{\mathbf{x}_{\text{new}}}^*}^*} \phi(\mathbf{z}, \xi_k) \right\}. \quad (11)$$

From a solution perspective, the training problem (7)–(10) is a non-convex mathematical program with a decision-tree-encoding constraint and a bi-level optimization component, challenging to be efficiently handled by standard optimization algorithms or off-the-shelf solvers. To tackle this issue, in the next section, we present an efficient recursive-partitioning greed heuristic strategy to tackle problem (7)–(10) as an extension of the standard Classification and Regression Trees (CART) [Breiman et al., 1984] leveraging upon the perturbation method described by [Murthy et al., 1994].

3. Solution Approach: A Recursive-Partitioning Heuristic Strategy

Problem (7)–(10) is a non-convex mathematical program whose solution capability by standard optimization algorithms and off-the-shelf solvers is reduced. Therefore, in this section, we adapt the structure of the recursive-partitioning algorithmic framework discussed in [Breiman et al., 1984; Bertsimas and Dunn, 2019] to the proposed predictive- and prescriptive-oriented training problem. Structurally, as discussed in [Bertsimas and Dunn, 2019], the hierarchical partition of a decision tree typically follows an affine rule strategy (hereinafter referred to as a *hyperplane split*) in which a linear function (hyperplane) of the covariates define the branching rule for partitioning, i.e., $h_\varphi(\mathbf{x}_k) = \sum_{i=1}^{m_x} \varphi_i x_{i,k} - t$. More specifically, let $x_{i,k} \in \mathcal{X}_i$ be the k -th observation of an i -th covariate component. Then, the proposed heuristic strategy seeks a partition of the covariate space as $\mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t) \leftarrow \{k \in \mathcal{K} \mid \sum_{i=1}^{m_x} \varphi_i x_{i,k} < t\}$ and $\mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t) \leftarrow \{k \in \mathcal{K} \mid \sum_{i=1}^{m_x} \varphi_i x_{i,k} \geq t\}$, that minimizes the following optimization problem:

$$\min_{\substack{\{\varphi_i\}_{i=1}^{m_x}, t, \\ \psi_{\text{left}} \in \Psi_{\text{left}}, \\ \psi_{\text{right}} \in \Psi_{\text{right}}}} \left\{ \sum_{k \in \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \Phi_\lambda \left(\arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{k \in \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \phi(\mathbf{z}, \xi_k) \right\}, \psi_{\text{left}}, \xi_k, \mathbf{x}_k \right) + \right. \\ \left. \sum_{k \in \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \Phi_\lambda \left(\arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{k \in \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \phi(\mathbf{z}, \xi_k) \right\}, \psi_{\text{right}}, \xi_k, \mathbf{x}_k \right) \right\}. \quad (12)$$

To reach this objective, we recognize that a greedy search in the space of all hyperplanes, selecting the one with the lowest objective value in (12), is computationally prohibitive. Thus, we leverage a perturbation approach as in [Murthy et al., 1994] to identify, for a given hyperplane, what is the least increase in the coefficient of a single covariate component that “moves” only one observation from the *left* to the *right* children node. The procedure thus “greedily” evaluates (12) for all possible single changes from left to right children node for all covariate components. To calculate this critical point, let a hyperplane split be given and defined by $(\{\varphi_i\}_{i=1}^{m_x}, t)$. A strictly non-zero observation $k \in \mathcal{K}$ will belong to the left

child if the coefficient of a covariate component $j \in \{1, \dots, m_x\}$ satisfies the following condition

$$\sum_{i=1}^{m_x} \varphi_i x_{i,k} < t \iff \varphi_j < \frac{1}{x_{j,k}} \left(t - \sum_{i \in \{1, \dots, m_x\} \mid i \neq j} \varphi_i x_{i,k} \right). \quad (13)$$

Additionally, we improve the splitting procedure by “removing” a covariate component from the hyperplane. This follow-on step potentially reduces the complexity of the resulting tree. For this purpose, for a given hyperplane split defined by $(\{\varphi_i\}_{i=1}^{m_x}, t)$, an observation $k \in \mathcal{K}$ will belong to the left child when removing a covariate component $j \in \{1, \dots, m_x\}$ (i.e., by setting $\varphi_j = 0$) if the following condition is satisfied

$$t > \sum_{i \in \{1, \dots, m_x\} \mid i \neq j} \varphi_i x_{i,k}. \quad (14)$$

Hence, the overall idea of the proposed heuristic strategy is to define coefficients of the hyperplane that sequentially “moves” one observation from the left to the right children node based on the bounds for the slope coefficients (13) and the intercept (14), then evaluate (12), and store the corresponding hyperplane split $(\{\varphi_i\}_{i=1}^{m_x}, t)$ that leads to the least value. It should be noted that the described procedure relies on a given hyperplane split already defined to then update its parameters using perturbations and deletion, seeking a local minimum. Due to the non-convexity of the problem, it might exist multiple local minima at a node in the tree, and so which minimum is reached depending on the given “starting” hyperplane parameters $(\{\varphi_i\}_{i=1}^{m_x}, t)$. We address this issue by considering S random hyperplane splits as starting points. After a *hyperplane split* is found, the partitioning is then recursively called in the left and right children nodes until a stopping criterion is met. For this heuristic strategy, following standard practices, we adopt maximum depth size ($DPH_{\max}$) for the tree and minimum number of observations per leaf ($N_{\min}$) as stopping criteria. The proposed heuristic strategy to tackle the proposed predictive- and prescriptive-oriented training problem (7)–(10) is described in **Algorithm 1**.

3.1. Extension to an Ensemble of Decision Trees

We also consider as a pipeline for prescription an ensemble of Decision Trees. In the past decades, researchers and practitioners have observed an increased value in the predictive capability by performing a bootstrap-aggregate (bagging) procedure considering a set of Decision Trees. This procedure has been called in the technical literature as a Random Forest and is known to have less variance. In the particular context of this work, the extension of the prescription framework presented in Section (2) goes as follows. Let E be the number of Decisions Trees the decision-maker is willing to consider to devise its prescription. Then, for each “Decision Tree” $e \in \{1, \dots, E\}$, sample without reposition (the bootstrap stage) a subset $\mathcal{K}_e \subset \mathcal{K}$ of historical data. Further on, train the Decision Tree solving problem (7)–(10) using **Algorithm 1**, but considering only the random subset $\mathcal{K}_e$. For expository purposes, we map the size of each subset of historical data as $|\mathcal{K}_e| = \lceil \gamma |\mathcal{K}| \rceil$ for some *a priori* defined $\gamma \in (0, 1]$. Finally, after training all E Decision Trees, for a given new covariate data $\mathbf{x}_{\text{new}}$, to assess the prescription, we perform a similar two-step procedure described in Section (2.1) but “aggregating” the E Decision Trees as follows:

$$\mathbf{z}^*(\mathbf{x}_{\text{new}}) \in \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{e=1}^E \frac{1}{\left| \{k \in \mathcal{K} \mid \mathbf{x}_k \in \mathcal{L}_{\mathbf{x}_{\text{new}}, e}^*\} \right|} \sum_{k \in \mathcal{K} \mid \mathbf{x}_k \in \mathcal{L}_{\mathbf{x}_{\text{new}}, e}^*} \phi(\mathbf{z}, \xi_k) \right\}, \quad (15)$$

with $\mathcal{L}_{\mathbf{x}_{\text{new}}, e}^*$ the associated leaf of the e -th trained Decision Tree such that the new covariate data $\mathbf{x}_{\text{new}}$ falls into.

4. Numerical Experiments

To illustrate the prescriptive capability of the proposed methodology, we present and discuss in this section a set of numerical experiments, comparing it to the following four typical benchmarks found in the technical literature for predictive distribution estimation:

Algorithm 1 Predictive- and Prescriptive-Oriented Training Algorithm

```

function Tree.HyperplaneSplit( $\{(\mathbf{x}_k, \boldsymbol{\xi}_k)\}_{k \in \mathcal{K}}$ ,  $N_{\min}$ ,  $DPH_{\max}$ ,  $dph$ ,  $\{\mathcal{L}_l\}_{l=1}^L$ ,  $S$ )

// Initialization //
Set  $\mathbb{L}^* \leftarrow +\infty$  &  $\mathcal{K}_{\text{left}}^* \leftarrow \emptyset$  &  $\mathcal{K}_{\text{right}}^* \leftarrow \emptyset$ ; ▷ Initialize best solution

// Check Feasibility / Stopping Criteria – Depth Size and Minimum Number of Observations //
if ( $dph \leq DPH_{\max}$  ||  $|\mathcal{K}|/2 < N_{\min}$ ) then
    Set  $L \leftarrow L + 1$  and  $\mathcal{L}_L \leftarrow \{\mathbf{x}_k\}_{k \in \mathcal{K}}$ ; ▷ Create a new leaf
    Return

// Greed Search //
for  $s \in \{1, \dots, S\}$  do
    sample( $\{\varphi_i\}_{i=1}^{m_x}, t$ ); ▷ Sample an Hyperplane Split ( $\{\varphi_i\}_{i=1}^{m_x}, t$ )
    for  $i \in \{1, \dots, m_x\}$  do
         $U_{i,k} \leftarrow \frac{1}{x_{i,k}} \left( t - \sum_{j \in \{1, \dots, m_x\} \mid j \neq i} \varphi_j x_{j,k} \right)$ , ▷  $\forall k \in \mathcal{K}$ 
    sort( $\{U_{i,k}\}_{k \in \mathcal{K}}$ ); ▷ Sort  $\{U_{i,k}\}_{k \in \mathcal{K}}$  in the ascending order

    for  $k \in \{1, \dots, |\mathcal{K}| - 1\}$  do
         $\varphi_i \leftarrow \frac{1}{2} (U_{i,k} + U_{i,k+1})$ 
         $\mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t) \leftarrow \{k \in \mathcal{K} \mid \sum_{i=1}^{m_x} \varphi_i x_{i,k} < t\}$ ; ▷ Observations in the left child
         $\mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t) \leftarrow \{k \in \mathcal{K} \mid \sum_{i=1}^{m_x} \varphi_i x_{i,k} \geq t\}$ ; ▷ Observations in the right child

        // Evaluate the Predictive- and Prescriptive-Oriented Loss Function for the Left and Right children Nodes //
         $\mathbb{L} \leftarrow \min_{\psi_{\text{left}} \in \Psi_{\text{left}}} \left\{ \sum_{k \in \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \Phi_{\lambda} \left( \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{k \in \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \phi(\mathbf{z}, \boldsymbol{\xi}_k) \right\}, \psi_{\text{left}}, \boldsymbol{\xi}_k, \mathbf{x}_k \right) \right\} +$ 
 $\min_{\psi_{\text{right}} \in \Psi_{\text{right}}} \left\{ \sum_{k \in \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \Phi_{\lambda} \left( \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{k \in \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \phi(\mathbf{z}, \boldsymbol{\xi}_k) \right\}, \psi_{\text{right}}, \boldsymbol{\xi}_k, \mathbf{x}_k \right) \right\}$ ;

        if ( $\mathbb{L} < \mathbb{L}_{\text{best}}$  ||  $|\mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)| \geq N_{\min}$  ||  $|\mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)| \geq N_{\min}$ ) then
             $\mathbb{L}^* \leftarrow \mathbb{L}$  &  $\mathcal{K}_{\text{left}}^* \leftarrow \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)$  &  $\mathcal{K}_{\text{right}}^* \leftarrow \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)$ ; ▷ Update current best solution

         $\hat{U}_{i,k} \leftarrow \sum_{j \in \{1, \dots, m_x\} \mid j \neq i} \varphi_j x_{j,k}$ , ▷  $\forall k \in \mathcal{K}$ 
    sort( $\{\hat{U}_{i,k}\}_{k \in \mathcal{K}}$ ); ▷ Sort  $\{\hat{U}_{i,k}\}_{k \in \mathcal{K}}$  in the ascending order

    for  $k \in \{1, \dots, |\mathcal{K}| - 1\}$  do
         $\varphi_i \leftarrow 0$  and  $t \leftarrow \frac{1}{2} (\hat{U}_{i,k} + \hat{U}_{i,k+1})$ ;
         $\mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t) \leftarrow \{k \in \mathcal{K} \mid \sum_{i=1}^{m_x} \varphi_i x_{i,k} < t\}$ ; ▷ Observations in the left child
         $\mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t) \leftarrow \{k \in \mathcal{K} \mid \sum_{i=1}^{m_x} \varphi_i x_{i,k} \geq t\}$ ; ▷ Observations in the right child

        // Evaluate the Predictive- and Prescriptive-Oriented Loss Function for the Left and Right children Nodes //
         $\mathbb{L} \leftarrow \min_{\psi_{\text{left}} \in \Psi_{\text{left}}} \left\{ \sum_{k \in \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \Phi_{\lambda} \left( \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{k \in \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \phi(\mathbf{z}, \boldsymbol{\xi}_k) \right\}, \psi_{\text{left}}, \boldsymbol{\xi}_k, \mathbf{x}_k \right) \right\} +$ 
 $\min_{\psi_{\text{right}} \in \Psi_{\text{right}}} \left\{ \sum_{k \in \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \Phi_{\lambda} \left( \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \sum_{k \in \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)} \phi(\mathbf{z}, \boldsymbol{\xi}_k) \right\}, \psi_{\text{right}}, \boldsymbol{\xi}_k, \mathbf{x}_k \right) \right\}$ ;

        if ( $\mathbb{L} < \mathbb{L}_{\text{best}}$  ||  $|\mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)| \geq N_{\min}$  ||  $|\mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)| \geq N_{\min}$ ) then
             $\mathbb{L}^* \leftarrow \mathbb{L}$  &  $\mathcal{K}_{\text{left}}^* \leftarrow \mathcal{K}_{\text{left}}(\{\varphi_i\}_{i=1}^{m_x}, t)$  &  $\mathcal{K}_{\text{right}}^* \leftarrow \mathcal{K}_{\text{right}}(\{\varphi_i\}_{i=1}^{m_x}, t)$ ; ▷ Update current best solution

// Recursive Call in the Left and Right Child //
Call Tree.HyperplaneSplit( $\{(\mathbf{x}_k, \boldsymbol{\xi}_k)\}_{k \in \mathcal{K}_{\text{left}}^*}$ ,  $N_{\min}$ ,  $DPH_{\max}$ ,  $(dph + 1)$ ,  $\{\mathcal{L}_l\}_{l=1}^L$ ,  $S$ );
Call Tree.HyperplaneSplit( $\{(\mathbf{x}_k, \boldsymbol{\xi}_k)\}_{k \in \mathcal{K}_{\text{right}}^*}$ ,  $N_{\min}$ ,  $DPH_{\max}$ ,  $(dph + 1)$ ,  $\{\mathcal{L}_l\}_{l=1}^L$ ,  $S$ );
    
```

1. k -nearest neighbors (kNN) [Altman, 1992], defining k as $\lfloor \sqrt{|\mathcal{K}|} \rfloor$;
2. Nadaraya-Watson Kernel Regression [Nadaraya, 1964; Watson, 1964];
3. Neural Network using the implementation on the package *Flux.jl*. We considered a Neural Network with one hidden layer with 10 neurons and ReLU as the activation function, trained for 500 epochs [Hornik et al., 1989];
4. Random Forests, predicting each uncertain variable separately and then prescribing a decision based on this prediction. For this benchmark, we leverage the package *DecisionTree.jl*. Some characteristics were static, such as $N_{\min} = 20$, $\gamma = 75\%$, $N_{\text{subfeatures}} = m_x/3$, $E = 100$ [Breiman, 2001].

To appropriately evaluate the quality of the methods considered in this work, we define a validation set $\mathcal{K}_v$ with size 200. We make use of the so-called Coefficient of Prescriptiveness (P) to compare methods. To construct the metric, let $\mathbf{z}_{\text{SAA}}^*$ to denote the solution of a purely data-driven SAA problem, i.e.,

$$\mathbf{z}_{\text{SAA}}^* \in \arg \min_{\mathbf{z} \in \mathcal{Z}} \left\{ \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \phi(\mathbf{z}, \boldsymbol{\xi}_k) \right\}. \quad (16)$$

Then, the Coefficient of Prescriptiveness (P) is a unit-less quantity bounded by 1 and given by

$$P = \frac{\sum_{k \in \mathcal{K}_v} \phi(\mathbf{z}_{\text{SAA}}^*, \boldsymbol{\xi}_k) - \sum_{k \in \mathcal{K}_v} \phi(\mathbf{z}^*(\mathbf{x}_{\text{new},k}), \boldsymbol{\xi}_k)}{\sum_{k \in \mathcal{K}_v} \phi(\mathbf{z}_{\text{SAA}}^*, \boldsymbol{\xi}_k) - \sum_{k \in \mathcal{K}_v} \min_{\mathbf{z} \in \mathcal{Z}} \{ \phi(\mathbf{z}, \boldsymbol{\xi}_k) \}}, \quad (17)$$

which compares the methods' prescriptive capability with the purely data-driven SAA problem, regularized by a decision under Perfect Information. In the next subsections, we present an analysis of three different test problems comparing the proposed methodology with the four aforementioned benchmarks.

4.1. Case Study 1: Two-Stage Shipment Planning Problem

The Two-Stage Shipment Planning Problem (TS-SPP) aims to define the production in the d_x available warehouses so that the d_y demands are guaranteed to be met. In the first stage, the amount of units to be produced and stocked in each warehouse $z_i \geq 0$ at a cost $p_1 > 0$ /unit is defined. In the second stage, demand $\boldsymbol{\xi}$ at each point on the supply chain network is observed, and the products are shipped at a cost $c_{i,j}$. Also, it is possible to produce more in the second stage at each warehouse, but at a cost $p_2 > p_1$. The formulation of the TS-SPP considered in this work is presented in (18) – (20).

$$\phi(\mathbf{z}, \tilde{\boldsymbol{\xi}}) = \sum_{i=1}^{d_z} p_1 \cdot z_i + \min_{\mathbf{y}_1, \mathbf{y}_2 \geq \mathbf{0}} \left\{ \sum_{i=1}^{d_z} p_2 \cdot y_{1,i} + \sum_{i=1}^{d_x} \sum_{j=1}^{d_y} c_{i,j} \cdot y_{2,i,j} \right\} \quad (18)$$

subject to:

$$\sum_{i=1}^{d_z} y_{2,i,j} \geq \tilde{\xi}_j, \quad \forall j \in \{1, \dots, d_y\}; \quad (19)$$

$$\sum_{j=1}^{d_y} y_{2,i,j} \leq z_i + y_{1,i}, \quad \forall i \in \{1, \dots, d_z\}. \quad (20)$$

We will focus on the best result for each method. The kNN method found good prescriptions, with $P = 0.288$, the best one for this case study. The Nadaraya-Watson Kernel Regression, the Neural Network and Random Forest did not perform well. The values of the coefficient were, respectively, $P = -0.23$, $P = -1.244$ and $P = -1.081$. Considering the proposed method, the coefficient did not indicate a gain compared to SAA, however, it also did not indicate a worse solution (as in the case of Kernel, NN and RF). Considering only one tree, the best coefficient is $P = -0.079$ and considering Random Forests $P = 0.029$. On table 1 there is a summary of results.

Table 1: Summary of coefficients for Case Study 1: Shipment Planning

Method	P	Characteristics
kNN	0.288	$\mathcal{K} = 500$
DF tree RF	0.029	$\mathcal{K} = 200, \lambda = 0.0, E = 5$
DF tree	-0.079	$\mathcal{K} = 50, \lambda = 0.0, DPH_{\max} = 3$
Kernel	-0.23	$\mathcal{K} = 200$
RF	-1.081	$\mathcal{K} = 500, DPH_{\max} = 3$
NN	-1.244	$\mathcal{K} = 200$

4.2. Case Study 2: Stochastic Shortest-Path Problem

The main objective of this problem is to define the least-cost shortest path, with uncertainty in the cost values. Formally, consider a graph $(\mathcal{N}, \mathcal{A})$, with N nodes, with n_s the start node of the path and n_t the end node. Furthermore, for a given $n \in \mathcal{N}$, let $\mathcal{N}_{\text{In}}(n)$ and $\mathcal{N}_{\text{Out}}(n)$ be the set of nodes whose arc enters and leaves n . Then, the formulation of the S-SPP considered in this work is presented in (21) – (22).

$$\phi(\mathbf{z}, \tilde{\xi}) = \sum_{(m,n) \in \mathcal{A}} \tilde{\xi}_{m,n} z_{m,n} \quad (21)$$

with

$$\mathcal{Z} = \left\{ \mathbf{z} \in \mathbb{R}_+^{|\mathcal{A}|} \left| \begin{array}{l} \sum_{n \in \mathcal{N}_{\text{Out}}(n_s)} z_{n_s, n} = 1; \\ \sum_{m \in \mathcal{N}_{\text{In}}(n_t)} z_{m, n_t} = -1; \\ \sum_{m \in \mathcal{N}_{\text{Out}}(n)} z_{n, m} = \sum_{m \in \mathcal{N}_{\text{In}}(n)} z_{m, n}, \quad \forall n \in \mathcal{N} \setminus \{n_s, n_t\}; \end{array} \right. \right\}. \quad (22)$$

The best results from each method are also shown. In this case, neither Nadaraya-Watson Kernel Regression nor Neural Network worked well, with $P = 0.0$ and $P = 0.046$, respectively. In its turn, Random Forest did a good prescription, with $P = 0.341$, and is the best benchmark result, as kNN had $P = 0.12$. Lastly, our proposal worked really well on this case study. When considering Random Forest, a $P = 0.141$ was found, and when considering only one tree a $P = 0.364$ was found, the best solution for this case study, as can be seen on Table 2.

Table 2: Summary of coefficients for Case Study 2: Shortest Path

Method	P	Characteristics
DF tree	0.364	$\mathcal{K} = 200, \lambda = 1.0$
RF	0.341	$\mathcal{K} = 100, DPH_{\max} = 3$
DF tree RF	0.141	$\mathcal{K} = 200, 500, \lambda = 1.0, E = 5$
kNN	0.120	$\mathcal{K} = 500$
NN	0.046	$\mathcal{K} = 500$
Kernel	0.0	$\mathcal{K} = 50, 200, 500$

4.3. Case Study 3: Two-Stage Farmer's Problem

In this test problem, a farmer cultivates N products in a farm that is limited by L (km^2). Each product $i \in \{1, \dots, N\}$ has a cost of production given by c_i , a selling price $p_i^{(1)}$ valid if the amount of products sold is up to a_i , and a selling price $p_i^{(2)}$, for the remaining products. On the other hand, if the production does not meet demand, then the farmer should buy at a price b_i . The TS-FM is formulated as in (23)–(25).

$$\phi(\mathbf{z}, \tilde{\xi}) = \sum_{i=1}^N c_i z_i + \min_{(y_1, y_2^{(1)}, y_2^{(2)}) \geq 0} \left\{ \sum_{i=1}^N \left(b_i y_{1,i} - p_i^{(1)} y_{2,i}^{(1)} - p_i^{(2)} y_{2,i}^{(2)} \right) \right\} \quad (23)$$

subject to:

$$y_{2,i}^{(1)} \leq a_i, \quad \forall i \in \{1, \dots, N\}; \quad (24)$$

$$\left. \tilde{\xi}_i z_i + y_{2,i}^{(1)} + y_{2,i}^{(2)} - y_{1,i} \geq h_i, \quad \forall i \in \{1, \dots, N\}; \right\}. \quad (25)$$

In the case of the benchmarks, kNN did not enhance the SAA quality, with $P = -0.027$. The Neural Network and Random Forest did not perform well, with a $P = -0.297$ and $P = -0.116$, respectively, and Nadaraya-Watson Kernel Regression had the best prescription, with $P = 0.165$. Not far from that is the proposed method considering Random Forests, with $P = 0.101$. When considering only one tree, the proposal performed like kNN, with $P = 0.031$, as seen on Table 3.

Table 3: Summary of coefficients for Case Study 3: Farmer's Problem

Method	P	Characteristics
Kernel	0.165	$\mathcal{K} = 100$
DF tree RF	0.101	$\mathcal{K} = 100, \lambda = 0.25, 0.50, 0.75, 1.0, E = 5$
DF tree	0.031	$\mathcal{K} = 200, \lambda = 0.25, 0.50, 0.75, 1.0$
kNN	0.027	$\mathcal{K} = 100$
RF	-0.116	$\mathcal{K} = 200, DPH_{\max} = 3$
NN	-0.297	$\mathcal{K} = 200$

4.4. Summary of Results

Related to the benchmark methods, some characteristics can be noted from the previous analysis. Firstly, it was observed that most Machine Learning methods work better on only one case study instead of having an overall quality between all of them. In the case of kNN, it found relevant enhancements compared to SAA in the Shipment Plan and Shortest Path Problems. At the same time, Nadaraya-Watson Kernel Regression and Random Forest only enhanced the results on Farmer's and Shortest Path Problems, respectively. When varying $\mathcal{K}$, just kNN showed that more train data leads to better results. Shipment Planning and Shortest Path quality notoriously enhance with the increase of $\mathcal{K}$.

The proposed method worked better for the Shortest Path Problem, even though when considering more than one tree, it also worked really well for the Farmer's and Shortest Path Problems. Furthermore, different from the benchmarks, when it did not perform better than SAA, it did not perform worse, maintaining the best Coefficient of Prescriptiveness always close to or superior to zero. Moreover, it is notorious the necessity of the decision-tree-based structure $\mathcal{R}$ to enhance SAA results. Also, there is a recurrence of the good results, as varying λ and $\mathcal{K}$ tended to maintain the good prescriptions.

5. Conclusions

In this work, we construct a novel methodology that uses covariate data to help decision-making. Historically, this problem is solved with a two-stage stochastic optimization framework, considering sample average approximation (SAA) techniques. This technique examines only the historical uncertain data to make a prescription. We applied the proposed methodology to three test problems. We found that the proposed methodology enhances the results, compared to sample average approximation (SAA), in two of the case studies.

References

- Altman, N. S. (1992). An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician*, 46:175–185.
- Bertsimas, D. and Dunn, J. (2017). Optimal Classification Trees. *Machine Learning*, 106(7):1039–1082.
- Bertsimas, D. and Dunn, J. (2019). *Machine Learning under a Modern Optimization Lens*. Dynamic Ideas LLC Charlestown, MA.

- Bertsimas, D. and Kallus, N. (2020). From Predictive to Prescriptive Analytics. *Management Science*, 66(3):1025–1044.
- Birge, J. R. and Louveaux, F. (1997). *Introduction to Stochastic Programming*. Springer-Verlag, New York, NY, USA.
- Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.
- Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Taylor & Francis Group, LLC. Chapter 10, pp. 279–294.
- Casella, G. and Berger, R. L. (2002). *Statistical Inference*. Thomson Learning Academic Resource Center, Boston, MA, USA, second edition.
- Christoffersen, P. F. and Diebold, F. X. (1997). Optimal Prediction Under Asymmetric Loss. *Econometric Theory*, 13(6):808–817.
- Elmachtoub, A. N. and Grigas, P. (2022). Smart “Predict, then Optimize”. *Management Science*, 68(1):9–26.
- Estes, A. S. and Richard, J.-P. P. (2023). Smart predict-then-optimize for two-stage linear programs with side information. *INFORMS Journal on Optimization*, 5(3):295–320.
- Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366. ISSN 0893-6080.
- James, G., Witten, D., Hastie, T., Tibshirani, R., and Taylor, J. (2023). *An Introduction to Statistical Learning: with Applications in Python*. Springer, Berlin, Germany, first edition.
- Kallus, N. and Mao, X. (2023). Stochastic optimization forests. *Management Science*, 69(4):1975–1994.
- Kannan, R., Bayraksan, G., and Luedtke, J. R. (2024). Residuals-based Distributionally Robust Optimization with Covariate Information. *to appear in Mathematical Programming*, p. 1–57.
- Lanchier, N. (2017). *Stochastic Modeling*. Springer, Berlin, Germany.
- Murthy, S. K., Kasif, S., and Salzberg, S. (1994). A System for Induction of Oblique Decision Trees. *Journal of Artificial Intelligence Research*, 2:1–32.
- Nadaraya, E. A. (1964). On Estimating Regression. *Theory of Probability & Its Applications*, 9(1):141–142.
- Parys, B. P. G. V., Esfahani, P. M., and Kuhn, D. (2021). From Data to Decisions: Distributionally Robust Optimization Is Optimal. *Management Science*, 67(6):3387–3402.
- Sen, P. K. and Singer, J. M. (2019). *Large Sample Methods in Statistics (1994): An Introduction with Applications*. CRC Press, Boca Raton, Florida, US.
- Shapiro, A., Dentcheva, D., and Ruszczyński, A. (2009). *Lectures on Stochastic Programming: Modeling and Theory*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, USA, first edition.
- Stratigakos, A., Camal, S., Michiorri, A., and Kariniotakis, G. (2022). Prescriptive trees for integrated forecasting and optimization applied in trading of renewable energy. *IEEE Transactions on Power Systems*, 37(6):4696–4708.
- Watson, G. S. (1964). Smooth Regression Analysis. *Sankhya: The Indian Journal of Statistics*, 26(4):359–372.