Constrained Random Forest for Domain-Generalizable Classification

Camilla Lingjærde [1] **Geir Kjetil Sandve** [2] **Arnoldo Frigessi**[3] **Sylvia Richardson**[3, 4] **Johan Pensar**[1]

[1]Department of Mathematics, University of Oslo, Oslo, Norway
[2]Department of Informatics, University of Oslo , Oslo, Norway
[3]Institute of Basic Medical Sciences, University of Oslo, Oslo, Norway
[4]MRC Biostatistics Unit , University of Cambridge , Cambridgeshire, UK

Abstract

Domain generalization is a critical challenge in machine learning, where models must generalize outside their training domain(s) to unseen test or deployment domains without retraining. Motivated by the principle of independent causal mechanisms, the notion of domain invariance provides a principled causality-inspired approach to domain generalization. In this context, we introduce a general domain-constrained supervised learning framework where the aim is to learn a representation of the input data that results in improved domain invariance and stability in the resulting prediction model. Adapting the framework to tree-based learning, we introduce the Constrained Random Forest that prioritizes stable predictive mechanisms, leading to improved generalization across diverse environments. Experiments on synthetic and real-world datasets show improved robustness over competing methods, even in settings where the underlying assumptions that enable domain-invariant learning in theory are violated.

1 INTRODUCTION

Supervised classification methods, and tree-based classifiers in particular, often achieve excellent performance when applied to data that follows the same distribution as the training data. In many real-life applications, this assumption is, however, fragile: training data might be collected across a limited set of environments (domains) such as hospitals, regions, or time periods, which naturally partitions the observations and induces systematic distributional differences; future data may arise from new, unobserved environments.

These shifts induce an out-of-distribution (OOD) generalization problem: robust classification should rely on predictive mechanisms that hold across environments, rather than environment-specific associations [Schölkopf et al., 2021]. This is generally not achieved with standard nonparametric learners such as random forests [Breiman, 2001], which tend to exploit any predictive signal available [Zhou et al., 2022, Wang et al., 2022].

A large body of work on domain generalization formalizes robustness through invariance [Kaddour et al., 2022]. While methods based on Invariant Risk Minimization [Arjovsky et al., 2019] have shown to be outperformed by standard empirical risk minimization [Rosenfeld et al., 2021] in many settings, several causality-inspired approaches have demonstrated improved generalization. Invariant Causal Prediction (ICP; [Peters et al., 2016]) seeks predictor sets for which the conditional distribution of the target is invariant across environments, motivating classifiers that restrict attention to invariant features. Related ideas appear in anchor regression and its boosting variants, which penalize sensitivity to an environment variable and have been adapted to classification [Rothenhäusler et al., 2021, Kook et al., 2022, Londschien et al., 2026]. In particular, Invariant Random Forests (IRF; [Liao et al., 2024]) build invariance into the tree growing procedure by modifying the splitting criterion. However, as we demonstrate in this paper, existing invariance-based procedures can struggle when subsets of features are both highly predictive and strongly environment-dependent. Greedy methods such as tree-based models, boosting, or neural networks are especially vulnerable: early splits or updates can implicitly partition the data by domain (i.e., *domain instability*), so that subsequent substructures are effectively trained on domain-specific subsets and thus learn environment-specific rather than invariant mechanisms.

In this paper, we take a structured approach to domain generalization. First, we give the definition of an *optimal domain-invariant feature set* and provide a graphical characterization of it in a simple causal setting. Second, motivated by this notion, we introduce the general framework of *domain-constrained learning*, which combines an invariance-style constraint in the spirit of invariant prediction [Peters et al., 2016, Rojas-Carulla et al., 2018, Magliacane et al., 2018,

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:3888–3921.

Liu et al., 2023b] with a complementary domain stability constraint, which is novel in the considered context and penalizes features acting primarily as environment markers [Ganin and Lempitsky, 2015]. Third, we adapt tree-based learning to this framework to obtain a *Constrained Random Forest* (CRF) for classification that directly targets domain separation as a distinct source of OOD failure in tree ensembles. To our knowledge, this is the first tree-based domain generalization method that explicitly prevents the early "environment-partitioning" behavior. Finally, we show improved OOD generalization of CRF on synthetic and benchmark tasks under substantial shifts.

2 PROBLEM STATEMENT

We consider the classification setting where the aim is to predict the class of a categorical response variable $Y \in \mathcal{Y}$ given some input variables $X = (X_1, \ldots, X_p) \in \mathcal{X} \subseteq \mathbb{R}^p$ across domains (or environments) defined by a categorical $Z \in \mathcal{Z}$. The domain labels are assumed to be known. In this setting, the joint distribution over (X, Y), denoted by P, can be viewed as a mixture distribution:

$$p(X, Y) = \sum_{z \in \mathcal{Z}} p(X, Y \,|\, Z = z) p(Z = z), \qquad (1)$$

consisting of domain-specific components:

$$p(Y, X \,|\, Z = z), \text{ for each } z \in \mathcal{Z},$$

with associated mixing weights such that $\sum_{z \in \mathcal{Z}} p(Z = z) = 1$. The goal is to find a model $f : \mathcal{X} \to \mathcal{Y}$ that generalizes across all domains even if not all of them are observed in the training distribution. Specifically, the training distribution, P_{train}, is also a mixture distribution:

$$p_{\text{train}}(X, Y) = \sum_{z \in \mathcal{Z}_{\text{train}}} p(X, Y \,|\, Z = z) p_{\text{train}}(Z = z), \quad (2)$$

with the same domain-specific component as above, but with modified mixing weights such that $\sum_{z \in \mathcal{Z}_{\text{train}}} p_{\text{train}}(Z = z) = 1$ and $\mathcal{Z}_{\text{train}} \subset \mathcal{Z}$, and where $z_{\text{test}} \in \mathcal{Z} \setminus \mathcal{Z}_{\text{train}}$.

2.1 AN OPTIMAL DOMAIN-INVARIANT SET

To achieve domain generalization in the considered setting, we build on the idea of Peters et al. [2016], developed in the context of Structural Causal Models (SCMs), and look for a subset of features $S \subseteq \{1, \ldots, p\}$ for which we have domain invariance in the conditional distribution:

$$p(Y \,|\, x_S, z) = p(Y \,|\, x_S),$$

for every $(x_S, z) \in \mathcal{X}_S \times \mathcal{Z}$ for which $p(x_S, z) > 0$. In other words, interventions on the features should not influence the conditional distribution of Y given X_S (see Figure 1); equivalently, the target response and the environment should be conditionally independent given the set of features w.r.t. P, as denoted by:

$$(Y \perp_P Z \mid X_S).$$

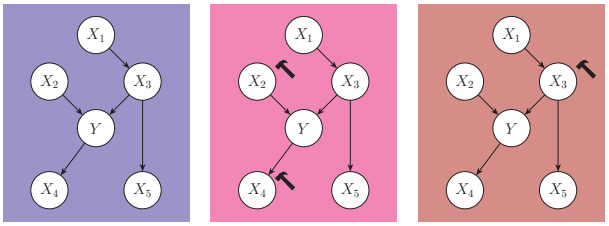


Figure 1: Example of an SCM with a categorical environmental variable corresponding to three different intervention distributions; (a) environment $Z = 1$ with no interventions, (b) environment $Z = 2$ with interventions on X_2 and X_4, and (c) environment $Z = 3$ with intervention on X_3. The invariance condition $(Y \perp_P Z \mid X_S)$ holds for the direct causes $X_S = \{X_2, X_3\}$ of Y, but will be violated if additionally including X_4 or if omitting either of X_2 and X_3. The figure is adapted from Peters et al. [2016].

Based on the above notion of invariance and in the context of prediction, we define an optimal domain-invariant subset $S^* \subseteq \{1, \ldots, p\}$ as a minimal subset that provides the most possible information about Y without violating the domain invariance criterion.

Definition 1 (Optimal domain-invariant set)**.** *Let Y be a categorical response variable, $X = (X_1, \ldots, X_p)$ a set of random feature variables, and Z a categorical domain variable. Furthermore, let P denote the joint distribution over (X, Y, Z). For the purpose of predicting Y from X, we define an optimal domain-invariant feature set, S^*, as the subset of features $S \subseteq \{1, \ldots, p\}$ for which the following independence statements hold:*

1. $(Y \perp_P Z \mid X_S)$ *(domain invariance),*
2. $(Y \not\perp_P X_W \mid X_{S \setminus W})$ *for any $W \subseteq S$ (minimality),*
3. $(Y \perp_P X_W \mid X_S)$ *or* $(Y \not\perp_P Z \mid X_{S \cup W})$ *for any $W \subseteq \{1, \ldots, p\} \setminus S$ (sufficiency).*

Informally, in the above definition: the first condition is the central invariance requirement in invariant prediction and much of the domain generalization literature [e.g. Peters et al., 2016, Rojas-Carulla et al., 2018, Magliacane et al., 2018, Liu et al., 2023b]; the second and third conditions are feature-set optimality refinements, enforcing minimality (no redundant features) and optimality (no additional predictive feature can be added without violating domain invariance), respectively. The associated conditional distribution $p(Y \mid X_{S^*})$ is an optimal prediction model that can be derived

under any P_{train}, as defined in (2) [1]. Assuming that X_{S^*} is identifiable from the (in)dependence relations implied by P_{train}, we are in theory able to recover the domain-invariant model which is optimal in the above sense without observing all domains.

Next, we present a motivation for the existence of an optimal domain-invariant set and a condition for its identifiability in the framework of Structural Causal Models (SCMs).

2.2 THE CAUSAL VIEW

We assume that the joint distribution P can be represented by an SCM, where the domain variable Z serves as a designated root node, similar to a confounder [Pearl, 2009].

Assumption 1. *The joint distribution P is induced by an SCM over $R = (X, Y, Z)$, such that the causal structure over R can be represented by a directed acyclic graph (DAG), G, where:*

$$\text{Pa}_{X_j} \subseteq R \setminus X_j, \ \text{Pa}_Y \subseteq R \setminus \{Y, Z\}, \ \text{and Pa}_Z = \varnothing.$$

Importantly, we assume that Z is not a direct cause of Y among the observed variables; otherwise one cannot, in general, obtain $Y \perp Z \mid X_S$. For now, no latent confounding is assumed. We next introduce the notion of *Y-faithfulness*, which is necessary for identifiability.

Definition 2 (Y-faithfulness)**.** *For the following set of conditional independence statements:*

$$\begin{aligned}\mathcal{I}_Y = &\{(Y \perp Z \mid W) : W \subseteq X\} \cup \\ &\{(Y \perp V \mid W) : V \subseteq X, W \subseteq X \setminus V\},\end{aligned}$$

let $\mathcal{I}_Y(P) \subseteq \mathcal{I}_Y$ and $\mathcal{I}_Y(G) \subseteq \mathcal{I}_Y$ denote the subset of CI statements induced by P and G, respectively. We say that P is Y-faithful to G if $\mathcal{I}_Y(P) \subseteq \mathcal{I}_Y(G)$.

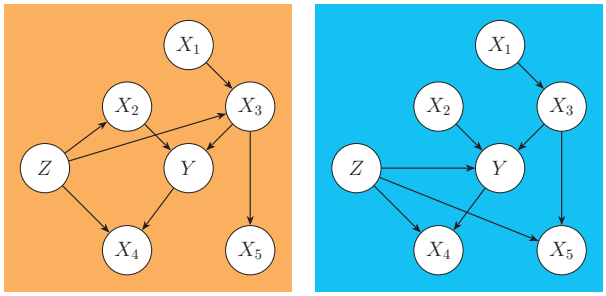
Note that faithfulness of the full distribution of the SCM to G implies Y-faithfulness, but the latter does not necessarily imply the former. Moreover, since P is induced by an SCM with DAG G, we have that $\mathcal{I}_Y(G) \subseteq \mathcal{I}_Y(P)$, which together with Y-faithfulness implies that $\mathcal{I}_Y(G) = \mathcal{I}_Y(P)$.

Assuming a DAG, the Markov blanket, which shields Y from the rest of the network, is given by:

$$\text{MB}_Y = \text{Pa}_Y \cup \text{Ch}_Y \cup \text{Sp}_Y,$$

where the sets on the right-hand side are known as the parents, children, and spouses of Y, respectively [Koller and Friedman, 2009]. We can then formally characterize the unique, optimal domain-invariant set from Definition 1 as a subset of the Markov blanket (see Figure 2 for an example), as anticipated by Rojas-Carulla et al. [2018].

[1] Its accuracy may still vary across domains due to shifts in $p(X_{S^*} \mid Z)$; equal accuracy across domains is only ensured when $Z \perp X_{S^*}$, which together with $Y \perp Z \mid X_{S^*}$ implies that $p(Y, X_{S^*} \mid Z = z) = p(Y, X_{S^*}), \ \forall z \in \mathcal{Z}$.



(a) (b)

Figure 2: Example of two SCMs with an environmental variable Z, where (a) is analogous to Figure 1. Consider the direct causes $X_S = \{X_2, X_3\}$ of Y. In (a), X_S is then the optimal domain-invariant set and $Y \perp Z \mid X_S$. In (b) there is no domain-invariant set
, but we have that $X_S \perp Z$.

Theorem 1 (Optimal domain-invariant set under a DAG)**.** *Let P be induced by an SCM with graph G, as described in Assumption 1, and assume that P is faithful to G. Then, the unique optimal domain-invariant set is given by:*

$$S^*_G = \text{Pa}_Y \cup \overline{\text{Ch}}_Y \cup \overline{\text{Sp}}_Y,$$

where

$$\begin{aligned}\overline{\text{Ch}}_Y &= \text{Ch}_Y \setminus \{\text{Ch}_Y \cap \text{Ch}_Z, \text{Desc}_{\text{Ch}_Y \cap \text{Ch}_Z}\}, \\ \overline{\text{Sp}}_Y &= \text{Sp}_Y \cap \text{Pa}_{\overline{\text{Ch}}_Y}.\end{aligned}$$

Proof. It is straightforward to show that S^*_G uniquely satisfies all three conditions in Definition 1. The full proof is given in Section A.1 in the Supplementary Material. □

In short, S^*_G in Theorem 1 captures the features in X that are predictive for Y without opening any paths between Y and Z. This is done by considering all X in Y's Markov Blanket, and excluding those that are: (i) shared children of Y and Z or descendants of the shared children, (ii) only included in the Markov Blanket by virtue of being pure spouses of the excluded children. While inclusion of (i) would violate the invariance condition in Definition 1 by opening a collider path between Y and Z, inclusion of (ii) would violate the minimality condition as these nodes are d-separated from Y when (i) is excluded from the conditioning set. We also note that if $\text{Ch}_Z = X$, then $S^*_G = \text{Pa}_Y$, that is, the direct causes, a fact that is exploited by ICP [Peters et al., 2016]. If $\text{Ch}_Y \cap \text{Ch}_Z = \varnothing$, then the optimal domain-invariant set is given by the Markov blanket, $S^*_G = \text{MB}_Y$, which was also identified as the optimal predictive set in the

single-domain setting by [Aliferis et al., 2010]. Moreover, the removal of $\mathrm{Desc}_{\mathrm{Ch}_Y \cap \mathrm{Ch}_Z}$ directly mirrors the graphical invariance condition of Liu et al. [2023a, Theorem 4.1], who characterize optimality via worst-case risk minimization.

While S^*_G in Theorem 1 is defined in terms of the global Markov properties of G (that is, $\mathcal{I}(G)$), under the assumption that P is faithful G, we have that $\mathcal{I}(P) = \mathcal{I}(G)$ and thus $S^*_G = S^*$. Moreover, in order to recover S^*_G based on a training distribution P_{train}, or in practice data generated from such a distribution, we need P_{train} to be faithful to the same causal structure, G. Or, more specifically, we need to be able to correctly infer the local conditional independence statements involving Y from P_{train}.

Corollary 1 (Identifiability of S^*_G)**.** *Let the joint distribution P be generated by an SCM, as described in Assumption 1, with causal structure G. Let P_{train} be the training distribution obtained from the same SCM as P, but with a modified distribution over variable Z; that is, $p_{\mathrm{train}}(x, y, z) = p_{\mathrm{train}}(z)\, p(x, y \mid z)$ where $p_{\mathrm{train}}(z) \geq 0$ and $\sum_{z \in \mathcal{Z}} p_{\mathrm{train}}(z) = 1$. Then, the unique optimal domain-invariant set $S^* = S^*_G$ is identifiable from P_{train}, if P_{train} is Y-faithful to G.*

Proof. The full proof is given in Section A.2 in the Supplementary Material. □

In practice, to detect a domain invariant set, one requires domain shifts that induce detectable changes in each $X_j \in \mathrm{Ch}_Z$ across some $z \neq z'$, so the relevant conditional independencies are identifiable from the training data. More generally, if the training environments are sufficiently diverse and the Y-specific CI relations can be estimated reliably, then S^* can be recovered from P_{train}.

Remark 1 - Domain Instability While Y-faithfulness assumes direct perturbations of every $X_j \in \mathrm{Ch}_Z$ in the training distribution, too strong perturbations may result in violations of this assumption and subsequent inclusion of variables that break the domain-invariance condition. In an extreme scenario, assume that X_{S^*} contains enough information to perfectly predict the domain label. Then, for every $x_{S^*} \in \mathcal{X}_{S^*}$, we have that

$$p(x_{S^*}, z) = p(z \,|\, x_{S^*}) p(x_{S^*}) = 0$$

for every domain label z except the correct one, implying that

$$p(Y \,|\, x_{S^*}, z) = p(Y \,|\, x_{S^*})$$

holds trivially for every $(x_{S^*}, z) \in \mathcal{X}_{S^*} \times \mathcal{Z}$ for which $p(x_{S^*}, z) > 0$, implying domain invariance. Then an issue will arise if there is a common child $X_j \in \mathrm{Ch}_Z \cap \mathrm{Ch}_Y$, since X_j will then be informative for Y in a predictive sense, and its inclusion will not violate the domain-invariance condition which will still trivially hold due to the above mentioned perfect separability of the domains Z given X_{S^*}, a property we refer to as domain instability. In practice, domain instability will be a real issue in the sense that already near-violations of faithfulness in the underlying training distribution might be emphasized in the finite training data. Moreover, greedy learners (e.g. trees, boosting, or neural networks) can amplify this effect: early splits or updates may implicitly partition the data by domain, so that subsequent components operate on domain-specific subsets. To counteract this, we will complement the notion of domain invariance with domain stability in the pursuit of domain-generalizable predictions in the next section.

Remark 2 - Latent Confounding Although our results assume no latent confounding, the general idea could in principle be extended to cover certain types of confounding structures. For example, consider a setting where the causal structure can be represented by a maximal ancestral graph (MAG: Richardson and Spirtes [2002]); under assumptions analogous to those of Assumption 1 ($Z \notin \mathrm{Pa}_Y$ and Z being a root node), a separating set between Z and Y is guaranteed to exist due to the maximality property of MAGs, thus ensuring the existence of a domain-invariant set. In more complex cases, domain invariance may not be achievable; for example, for $Z \to X \to Y$ with a latent confounder $X \leftarrow U \to Y$, we have that Z and Y are structurally dependent regardless of whether we condition on X or not, meaning that no domain-invariant set exists.

3 DOMAIN-CONSTRAINED LEARNING

We seek to learn a classifier that generalizes across environments $Z \in \mathcal{Z}$ by relying on predictive information from a subset $S \subseteq \{1, \ldots, p\}$ of features that supports invariant prediction while not acting as a strong environment marker. We therefore consider two constraints stated in terms of X_S:

$$\text{(i) Domain invariance:} \quad Y \perp Z \mid X_S,$$
$$\text{(ii) Domain stability:} \quad X_S \perp Z.$$

The two constraints are illustrated with SCM examples in Figure 2(a) and (b), respectively. Constraint (i) was, as discussed in Section 2.1, proposed by Peters et al. [2016] in the context of causal discovery under the SCM framework. Constraint (ii) is related to the domain adaptation approach of Ganin and Lempitsky [2015], who considered the setting where one has access to labeled data from a training domain and unlabelled data from a target test domain. The constraint is also related to *multi-task learning* [Caruana, 1997] where the environment can be considered an *unrelated task* [Paredes et al., 2012]; in short, it states that X_S should be uninformative for predicting to which domain an observation belongs.

In general, constraint (i) can be considered the main constraint, while constraint (ii) serves as an auxiliary constraint that enables constraint (i) to be expressed in a meaningful

way (See Remark 1). However, note that the two constraints will often be in conflict with each other (as in Figure 2). In fact, both constraints can be satisfied simultaneously only in scenarios where Z is not an ancestor of Y. Thus, while enforcing constraint (ii) as a hard restriction would guarantee non-separation of the domains, it could directly prevent the model from satisfying constraint (i). Similarly, enforcing constraint (i) as a hard restriction would not only lead to the issue of domain separation (as explained in Remark 1) but also be overly restrictive, particularly in real-life settings where assumptions only hold approximately. We therefore impose the constraints softly through two penalty terms in the objective function, allowing the method to balance domain stability against invariant predictive accuracy.

3.1 POPULATION-LEVEL SETTING

Using information-theoretic measures, we will consider the following domain-constrained objective at the population level:

$$H(Y \,|\, X_S) + \lambda_1 I(Y; Z \,|\, X_S) + \lambda_2 I(Z; X_S), \quad (3)$$

where $H()$ and $I()$ refer to the conditional entropy and mutual information, respectively, under the induced $p(Y, Z \,|\, X_S)$. Briefly, $H(Y \,|\, X_S)$ measures the classification error of our main target, while $I(Y; Z \,|\, X_S)$ and $I(Z; X_S)$ measure how much constraints (i) and (ii), respectively, are violated. In addition, $\lambda_1, \lambda_2 \geq 0$ are tuning parameters that control how strongly each constraint is enforced, and the tradeoff between them in the case of conflicts. Finally, the goal is to find the (minimal) subset of features, X_S, that minimizes (3). Alternatively, given that

$$I(Z; X_S) = H(Z) - H(Z \,|\, X_S),$$

we can re-express (3) as

$$H(Y \,|\, X_S) + \lambda_1 I(Y; Z \,|\, X_S) - \lambda_2 H(Z \,|\, X_S), \quad (4)$$

where the last component now directly measures the predictive accuracy of the domain variable, favoring a high entropy or, equivalently, poor predictability.

3.2 SAMPLE-LEVEL SETTING

Now, consider the sample-level setting where we have a set of data containing N joint observations, (x_i, y_i, z_i). Based on the data, we will estimate $p(Y, Z \,|\, X_S)$ using a model of the form $p_\theta(Y, Z \,|\, h(X))$, where θ defines a statistical model given a (possibly complex) feature representation $h(X)$, covering also the special case of feature selection when $h(X) = X_S$. In particular, let $\hat{\theta}$ be the maximum likelihood estimate under a given $h(X)$:

$$\hat{\theta} = \arg\max_{\theta} \left\{ \frac{1}{n} \sum_{i=1}^{n} \log p_\theta(y_i, z_i \,|\, h(x_i)) \right\}. \quad (5)$$

Then, our sample-level variant of (4) can be expressed as

$$\hat{H}(Y \,|\, h(X)) + \lambda_1 \hat{I}(Y; Z \,|\, h(X)) - \lambda_2 \hat{H}(Z \,|\, h(X)), \quad (6)$$

where the above measures are estimated based on the estimated model $p_{\hat{\theta}}(Y, Z \,|\, h(X))$ and the empirical measure $\hat{p}(h(X)) = P_n(h(X))$ (that is, by averaging over the data). Then, the general goal is to construct an efficient representation $h(X)$ that minimizes (6).

4 CONSTRAINED TREE LEARNING

In this work we implement the approach of Section 3.2 in the context of tree-based learning, where the representation of X is defined by a tree $\mathcal{T}$. Specifically, the tree defines a recursive partition of the input space, $\{R_m\}_{m=1}^{M}$, such that each input x is assigned to a class in the partition (or leaf). Accordingly, we let $h_\mathcal{T}(X) = m$ if $X \in R_m$. Associated to each leaf m, there will be a categorical distribution over (Y, Z), parameterized by θ_m. That is, θ_m contains the joint event probability $\theta_{y,z|m}$ for every $(y, z) \in \mathcal{Y} \times \mathcal{Z}$, which is simply estimated by the relative frequency:

$$\hat{\theta}_{y,z|m} = \frac{N_{y,z|m}}{N_m},$$

where $N_{y,z|m}$ denotes the number of observations with configuration (y, z) in leaf m and N_m is the total number of observations in leaf m. Thus, for a given $h_\mathcal{T}(X)$, the locally constant model can be expressed and estimated by:

$$p_{\hat{\theta}}(Y, Z \,|\, h_\mathcal{T}(X)) = \sum_{m=1}^{M} \hat{\theta}_{y,z|m} I(h_\mathcal{T}(X) = m),$$

meaning that the prediction depends on x only through the discrete representation $h_\mathcal{T}(X)$, greatly facilitating the evaluation of (6) while learning the representation.

4.1 CONSTRAINED DECISION TREES

In our constrained version of decision tree learning, we grow $\mathcal{T}$ by minimizing (6). This can effectively be viewed as minimizing standard entropy-based leaf impurity:

$$\hat{H}(Y \,|\, h_\mathcal{T}(X)) = -\sum_{m=1}^{M} \frac{N_m}{N} \sum_{y \in \mathcal{Y}} \hat{\theta}_{y|m} \log \hat{\theta}_{y|m},$$

with additional penalties that measures violations to (i) domain invariance:

$$\hat{I}(Y; Z \,|\, h_\mathcal{T}(X)) = \sum_{m=1}^{M} \frac{N_m}{N} \sum_{y \in \mathcal{Y}} \sum_{z \in \mathcal{Z}} \hat{\theta}_{y,z|m} \log \frac{\hat{\theta}_{y,z|m}}{\hat{\theta}_{y|m} \hat{\theta}_{z|m}}$$

and (ii) domain stability:

$$-\hat{H}(Z \,|\, h_\mathcal{T}(X)) = \sum_{m=1}^{M} \frac{N_m}{N} \sum_{z \in \mathcal{Z}} \hat{\theta}_{z|m} \log \hat{\theta}_{z|m}.$$

Since each of the above measures is leaf-decomposable, our learning objective, denoted by $\mathcal{L}(\mathcal{T}; \lambda_1, \lambda_2)$, is also leaf-decomposable:

$$\mathcal{L}(\mathcal{T}; \lambda_1, \lambda_2) = \sum_{m=1}^{M} \frac{N_m}{N} g(m), \tag{7}$$

where

$$g(m) = \hat{H}(Y \,|\, m) + \lambda_1 \, \hat{I}(Y; Z \,|\, m) - \lambda_2 \, \hat{H}(Z \,|\, m)$$

is simply composed of the terms obtained when conditioning on $h_{\mathcal{T}}(X) = m$.

When evaluating (7) it can be written as a weighted sum of leaf-local quantities, each computed from the empirical (Y, Z) contingency table within each leaf (details in Section B.1 in the Supplementary Material). Consequently, when a candidate split refines a node t, the change in (7) can be computed using only the node-local counts $\{N_{y,z|t}\}$ and those of its children, enabling a standard greedy tree-growing procedure. Specifically, let t denote a node during the tree growing process or, equivalently, a class in the intermediate recursive partition. Then, if a split refines t into children t_L, t_R, then the split changes (7) only locally by

$$\Delta\mathcal{L}_{t \to t_L, t_R} = \frac{N_{t_L}}{N} g(t_L) + \frac{N_{t_R}}{N} g(t_R) - \frac{N_t}{N} g(t).$$

We select the split minimizing $\Delta\mathcal{L}$ (equivalently maximizing the constrained gain $-\Delta\mathcal{L}$), subject to standard stopping rules and an ε-improvement threshold. As in standard tree-based approaches, this greedy split rule does not guarantee a global optimum of (7); more global tree optimization could in principle be considered, but would computationally be much more demanding. Further details of the Constrained Decision Tree and its implementation are given in Section B.1 in the Supplementary Material.

One practical aspect is worth highlighting. The penalty parameters λ_1 and λ_2 control the accuracy-invariance trade-off: when more than two training environments are available they can in principle be selected by cross-validation over an OOD-motivated split of environments. When only two training environments are available, however, such tuning is generally infeasible. Further, even with more environments, limited between-domain heterogeneity may lead cross-validation to under-select the penalties in favor of training accuracy. We therefore view fully data-driven selection of λ_1 and λ_2 as an open practical problem. Encouragingly, our empirical results suggest that performance is robust once both penalties are chosen sufficiently large, consistently improving OOD AUC and accuracy relative to competing baselines; in practice, we suggest $\lambda_1 = \lambda_2 = 10$ (unless $|\mathcal{Y}|$ and $|\mathcal{Z}|$ are substantially different; see Section B.1.4 in the Supplementary Material).

4.2 CONSTRAINED RANDOM FOREST

A constrained random forest (CRF) aggregates B constrained trees $\{\mathcal{T}^{(b)}\}_{b=1}^{B}$ grown under (7) with bootstrap aggregation (bagging). For prediction, we average the class-probabilities over the bagged tree models:

$$\hat{p}_{\mathrm{RF}}(y \,|\, x) = \frac{1}{B} \sum_{b=1}^{B} p_{\hat{\theta}^{(b)}}(y \,|\, h_{\mathcal{T}^{(b)}}(x) = m^{(b)}), \;\; y \in \mathcal{Y},$$

and assign the class label by

$$\hat{y}(x) = \arg\max_{y} \{\hat{p}_{\mathrm{RF}}(y \mid x)\}.$$

Since the constraints act at split selection, they shape the partition structure of each tree, and the ensemble inherits their favoring of invariant and domain-stable decision rules. Similarly to the IRF of Liao et al. [2024], we do not use random feature subsampling in CRF, since restricting the candidate set can force early splits to be chosen among sub-optimal variables for the constraints, weakening the intended control of domain separation and increasing the variability of invariance penalties across trees (see Section E.3 in the Supplementary Material). Further details of the Constrained Random Forest and its implementation are given in Section B.2 in the Supplementary Material.

5 EXPERIMENTS

5.1 SYNTHETIC DATA EXPERIMENTS

We study out-of-distribution (OOD) prediction under environmental shifts. In all synthetic experiments we generate data from a structural causal model (SCM) with binary outcome Y, observed covariates $X = (X_1, \ldots, X_p)$, and an observed environment indicator Z. We evaluate the performance of our and competing methods when trained on data from two source environments, and applied to a held-out target environment with a distributional shift induced by Z. Full details of the simulations, including SCM equations, parameters, and method hyperparameters are provided in Section C in the Supplementary Material.

We consider two simulation study setups. In the first setting we have $p = 5$ features that correspond directly with the nodes in Figure 2. We consider $n = 300$ respective training and test samples with environment and class balance, and $N = 100$ simulation replications. In addition, we consider a scaled-up version with $p = 25$ obtained by replacing each of the five features by a block of features and scaling edge weights with block size so that the magnitudes remain comparable across dimensions. In the scaled-up setting, we consider $n = 800$ respective training and test samples, and $N = 10$ simulation replications.

For both regimes, we consider five different variants of Figure 2 (models (a)-(e)), with the same nodes but different

causal structures. Briefly: model (a) is a classical invariance setting where constraint (i) holds for the predictive features $X_S = \{X_2, X_3\}$ (i.e., Figure 2 (a)); model (b) has a direct $Z \to Y$ effect which violates constraint (i) but constraint (ii) holds for $X_S = \{X_2, X_3\}$ (i.e., Figure 2 (b)); model (c) removes this $Z \to Y$ so that both constraints hold; model (d) adds a weak $Z \to Y$ effect to the structure of (a) so that neither constraint holds; and model (e) matches (a) but with a larger distributional shift in the test environment. Three of these models pose a particular challenge; in models (a), (d), and (e), X_2 is both strongly correlated with Z and highly predictive for Y. Greedy unconstrained trees may therefore split early on X_2, effectively separating the environments into near single-domain sub-trees; within such regions the Y–Z collider X_4 represents a domain-specific mechanism that is highly predictive for Y and thus a likely candidate for further splitting, despite having a negative impact on domain generalizability. To assess whether the methods successfully avoid domain separation and the resulting inclusion of collider X_4, we also report feature-importance summaries (computed as detailed in Section C.1.1 in the Supplementary Material).

We compare our constrained random forest (CRF) to standard random forest (RF), ICP feature selection combined with CART or RF (ICP+rpart, ICP+RF), Anchor Boosting, and IRF. For CRF we consider a small grid of penalty parameter pairs (λ_1, λ_2) (but limit it to $\lambda_1 = \lambda_2 = 10$ when aggregating results). Following the respective simulation study setups of Londschien et al. [2026] and Liao et al. [2024], we consider similar hyperparameter grids for Anchor Boosting and IRF. All remaining method-specific hyperparameters are fixed according to recommended or standard choices, with details given in Section C in the Supplementary Material. R code for our CRF implementation, and code for reproducing all results, is available at `https://github.com/Camiling/CRF_code`.

In the setting with $p = 5$, Table 1 shows that CRF gives the best OOD performance under model (a), with particularly strong test accuracy. Its importance profile concentrates on the invariant predictors $\{X_2, X_3\}$ and assigns essentially no weight to X_4. Unconstrained RF has near perfect training data accuracy but degrades OOD; in this regime with few features it often avoids X_4 (likely due to feature subsampling), yet its test accuracy still drops under the shift. ICP+rpart and ICP+RF also underperform OOD - especially ICP+rpart, which places substantial importance on X_4 - and we observe similar behaviour for IRF and Anchor Boosting. Overall, these patterns are consistent with sensitivity to domain separation, whereas CRF explicitly discourages it. It is notable that ICP, Anchor Boosting and IRF demonstrate this behavior in model (a), a particularly classical invariance setting Peters et al. [2016]. Section E in the Supplementary Material further shows the best performance with CRF is achieved when λ_1 and λ_2 are of comparable magnitude and jointly large, both for the constrained tree and the constrained forest, with the ensemble performing best overall.

Table 2 aggregates test performance across models (a)-(e) for $p = 5$. CRF achieves the best (or near-best) performance across all SCMs, including in the settings where the invariance assumption is violated (the direct edge $Z \to Y$ in models (b) and (d)). The largest relative gains appear in the more challenging domain shift models (a), (d), and (e), where the methods without explicit control of environment dependence degrade substantially. While Anchor Boosting demonstrates lower OOD generalizability across models, IRF is often competitive in AUC, but tends to yield lower test accuracy than CRF in the harder shift settings, suggesting that the classification rule of IRF might need re-calibration when applied to a new domain (assuming that labeled samples from the new domain are available).

In the setting with $p = 25$, Table 3 broadly reflects the earlier results: CRF maintains high OOD performance, being either the best or statistically indistinguishable from the best method across the SCMs. The main difference is that standard RF no longer manages to avoid splitting on X_4 - with many correlated variants, feature subsampling makes the collider features hard to exclude - and its OOD performance drops accordingly, especially in (a), (d), and (e).

5.2 REAL-DATA EXPERIMENTS

We evaluate the Constrained Random Forest (CRF) on six real-data classification tasks that have been highlighted as challenging OOD generalization problems in prior work [Gardner et al., 2023, Liu et al., 2023a, Liao et al., 2024]. We train on a small set of environments and evaluate on a held-out environment to induce distribution shifts. The six tasks are: three American Community Survey (ACS) prediction problems based on U.S. Census microdata [Ding et al., 2021] - Income (predicting whether an individual's income exceeds a threshold), Mobility (predicting whether an individual resides at the same address as one year ago), and Public Coverage (predicting whether an individual has public health insurance) - where we generalize across U.S. states (Z = state); a Diabetes Re-admissions task [Strack et al., 2014] predicting whether a patient is readmitted (Z = admission source); a US Accidents task [Moosavi et al., 2019] predicting whether an accident is severe (binary severity threshold) (Z = state); and a Financial Indicators task [Carbone, 2019] predicting whether the price of a stock increases over the next year from financial indicators (Z = year). Details of dataset access, domain splits, and pre-processing are given in Section D in the Supplementary Material.

Table 4 reports OOD accuracy. CRF achieves the best accuracy on four of the six tasks, outperforming IRF on all data sets and outperforming ICP with RF and Anchor Boosting

Table 1: Average results over $N = 100$ replicates, for data simulated from model (a) with $p = 5$ features. The table shows the train and test AUC and prediction accuracy, with all values within two standard errors of the best test data value in bold.

			Training		Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CRF	1	1	1.000 (0.000)	0.990 (0.007)	0.974 (0.019)	0.839 (0.062)	0.16	0.95	0.84	0.00	0.45
CRF	5	5	0.997 (0.002)	0.970 (0.011)	0.985 (0.012)	0.878 (0.053)	0.06	1.00	0.24	0.00	0.14
CRF	10	10	0.995 (0.003)	0.958 (0.012)	**0.987 (0.008)**	**0.897 (0.034)**	0.02	1.00	0.12	0.00	0.07
RF	-	-	1.000 (0.000)	1.000 (0.001)	0.966 (0.024)	0.750 (0.088)	0.00	0.98	0.60	0.07	0.64
ICP + rpart	-	-	0.996 (0.006)	0.994 (0.009)	0.569 (0.114)	0.563 (0.114)	0.08	0.73	0.94	0.40	0.92
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	0.970 (0.022)	0.757 (0.081)	0.04	0.98	0.63	0.01	0.64
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	0.872 (0.124)	0.545 (0.090)	0.00	0.91	0.32	0.51	0.25
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	0.886 (0.079)	0.539 (0.097)	0.00	0.98	0.03	0.10	0.01
AnchorBoosting	10	-	0.993 (0.075)	0.996 (0.043)	0.846 (0.141)	0.533 (0.094)	0.00	0.98	0.01	0.08	0.01
IRF	1	-	0.944 (0.010)	0.856 (0.018)	0.957 (0.031)	0.663 (0.063)	0.28	0.00	0.99	0.22	0.83
IRF	5	-	0.943 (0.010)	0.854 (0.018)	0.962 (0.029)	0.640 (0.060)	0.33	0.00	1.00	0.34	0.79
IRF	10	-	0.942 (0.010)	0.853 (0.019)	0.963 (0.027)	0.631 (0.059)	0.35	0.00	1.00	0.38	0.78

Table 2: Average test data results over $N = 100$ replicates, for data simulated from models (a)-(e) with $p = 5$ features. The table shows the AUC and prediction accuracy, with all values within two standard errors of the best value in bold.

			Model (a)		Model (b)		Model (c)		Model (d)		Model (e)	
Method	λ_1	λ_2	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy
CRF	10	10	**0.987**	**0.897**	0.989	**0.937**	**0.995**	**0.964**	**0.987**	**0.895**	**0.852**	**0.620**
RF	-	-	0.966	0.750	**0.994**	**0.934**	0.994	0.934	0.964	0.750	0.756	0.552
ICP + RF	-	-	0.970	0.757	**0.993**	0.881	0.994	0.882	0.970	0.758	0.788	0.553
AnchorBoosting	1	-	0.872	0.545	0.993	0.648	0.994	0.641	0.866	0.538	0.650	0.506
AnchorBoosting	5	-	0.886	0.539	**0.993**	0.662	0.994	0.663	0.869	0.531	0.657	0.506
AnchorBoosting	10	-	0.846	0.533	**0.993**	0.667	0.993	0.649	0.856	0.529	0.626	0.504
IRF	1	-	0.957	0.663	0.992	**0.935**	0.992	0.939	0.958	0.659	0.768	0.527
IRF	5	-	0.962	0.640	0.992	0.930	0.992	0.937	0.962	0.636	0.810	0.521
IRF	10	-	0.963	0.631	0.992	0.928	0.993	0.935	0.963	0.628	0.824	0.519

Table 3: Average test data results over $N = 10$ replicates, for data simulated from the scaled up version of models (a)-(e) with $p = 25$ features. AUC and prediction accuracy is shown, with values within two standard errors of the best in bold.

			Model (a)		Model (b)		Model (c)		Model (d)		Model (e)	
Method	λ_1	λ_2	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy
CRF	10	10	**0.994**	**0.927**	**0.997**	**0.966**	**0.997**	**0.971**	**0.994**	**0.925**	**0.929**	**0.780**
RF	-	-	0.923	0.595	0.909	0.682	0.953	0.688	0.922	0.606	0.818	0.547
ICP + RF	-	-	0.547	0.492	0.976	0.925	0.976	0.923	0.546	0.494	0.522	0.492
AnchorBoosting	1	-	0.546	0.534	0.994	0.509	0.986	0.508	0.539	0.537	0.507	0.512
AnchorBoosting	5	-	0.708	0.528	0.993	0.511	0.984	0.508	0.709	0.530	0.647	0.513
AnchorBoosting	10	-	0.815	0.514	0.989	0.544	0.985	0.508	0.763	0.517	0.717	0.503
IRF	1	-	**0.992**	0.742	**0.997**	0.955	**0.997**	0.963	0.991	0.744	**0.920**	0.639
IRF	5	-	0.989	0.721	**0.997**	0.958	**0.998**	0.967	0.988	0.724	**0.929**	0.622
IRF	10	-	0.987	0.686	**0.997**	0.956	**0.998**	0.966	0.985	0.689	**0.934**	0.603

Table 4: Accuracy of real-data classification tasks (shown as a percentage), with the best value achieved for each dataset in bold and standard errors in brackets.

Method	λ_1	λ_2	ACS Income	ACS Mobility	ACS Public Coverage	Diabetes Re-Admissions	US Accidents	Financial
CRF	10	10	**90.00** (1.73)	62.00 (2.43)	**66.50** (3.35)	58.40 (2.21)	**62.50** (3.43)	**63.30** (1.71)
RF	-	-	86.67 (1.97)	**66.75** (2.36)	62.50 (3.43)	58.00 (2.21)	57.50 (3.50)	57.12 (1.76)
ICP + RF	-	-	85.67 (2.03)	63.25 (2.41)	61.50 (3.45)	56.00 (2.22)	61.50 (3.45)	56.37 (1.76)
AnchorBoosting	1	-	77.67 (2.41)	65.50 (2.38)	63.00 (3.42)	59.00 (2.20)	57.50 (3.50)	59.77 (1.74)
AnchorBoosting	5	-	74.33 (2.53)	64.50 (2.40)	63.00 (3.42)	58.00 (2.21)	58.00 (3.50)	59.65 (1.74)
AnchorBoosting	10	-	79.67 (2.33)	63.50 (2.41)	63.00 (3.42)	**59.20** (2.20)	59.00 (3.49)	60.15 (1.74)
IRF	1	-	88.00 (1.88)	59.25 (2.46)	60.00 (3.47)	57.20 (2.21)	59.50 (3.48)	58.13 (1.75)
IRF	5	-	88.00 (1.88)	57.50 (2.47)	62.50 (3.43)	57.60 (2.21)	60.00 (3.47)	60.28 (1.74)
IRF	10	-	88.00 (1.88)	57.00 (2.48)	63.50 (3.41)	56.80 (2.22)	59.00 (3.49)	60.03 (1.74)

on five out of six. The differences in accuracy are, however, not statistically significant; apart from ACS Income, the absolute accuracies indicate that these are difficult OOD classification problems. This may be an indication of the environment shifts altering the outcome Y through unmeasured mechanisms (e.g., $Z - Y$ mediators or confounders), which could potentially be accounted for through domain adaptation if labeled test data is available. Nevertheless, the real-data classification tasks echo the synthetic data findings; CRF achieves competitive performance even when the causality assumptions motivating invariance are likely violated.

6 CONCLUSION

We have introduced a general framework of domain-constrained learning for the task of OOD generalization, by combining a causality-inspired domain invariance constraint with a domain stability constraint inspired by multi-task learning in the specific context of domain adaptation. Our proposed objective directly targets domain instability as a distinct and practically important source of OOD failure. We have adapted tree-based classifiers to this framework to obtain the Constrained Random Forest, for which the tree objective penalizes environment-dependent predictions within leaves and splits on features that primarily act as environment markers. Empirically, the Constrained Random Forest matches existing invariance-based competitors in settings where the invariant structure is comparatively easy to recover, while offering gains in settings where early environment partitioning is likely and in regimes where the invariance assumption is violated. Our experiments indicate that the domain stability constraint is particularly valuable for preventing strongly environment-dependent predictors from inducing near single-environment sub-trees and hence, more likely, non-generalizable decision rules.

Several practical extensions are natural. The constrained objective can be integrated into boosting-style tree ensembles and gradient-based learning methods, where the risk of compounding domain separation across stages makes stability control especially relevant. However, extending the framework to gradient-based learners such as deep neural networks is not straightforward. In that setting, the information-theoretic terms in the constrained objective function would need to be replaced or approximated for continuous learned representations rather than computed directly from discrete empirical distributions. One possible direction would be to replace these terms by predictive or entropy-based surrogate objectives, for example based on auxiliary models for the target and domain variables given the learned representation. Also, extending the framework to regression would broaden the applicability, yet it will require alternative dependence measures and loss decompositions suited to continuous targets. Finally, selecting λ_1 and λ_2 remains an open practical issue, especially with only two domains or limited between-domain variability where cross-validation may underselect penalties; encouragingly, our results suggest that performance is robust once both penalties are sufficiently large, yielding consistent improvements over competing methods across a wide range of OOD scenarios.

Acknowledgements

This has received funding from the European Union's Horizon Europe research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 101126636. This work was also supported by the Research Council of Norway, Integreat - Norwegian Centre for knowledge-driven machine learning, project number 332645.

References

Constantin F Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D Koutsoukos. Local causal and Markov blanket induction for causal discovery and feature selection for classification part I: Algorithms and empirical evaluation. *Journal of Machine Learning Research*, 11(7):171–234, 2010.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Leo Breiman. Random forests. *Machine Learning*, 45(1): 5–32, 2001.

Nicolas Carbone. 200+ financial indicators of us stocks (2014-2018), 2019. URL `https://www.kaggle.com/datasets/cnic92/200-financial-indicators-of-us-stocks-20142018/data`.

Rich Caruana. Multitask learning. *Machine Learning*, 28: 41–75, 1997.

Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing systems*, 34:6478–6490, 2021.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, volume 37, pages 1180–1189. PMLR, 2015.

Josh Gardner, Zoran Popovic, and Ludwig Schmidt. Benchmarking distribution shift in tabular data with tableshift. *Advances in Neural Information Processing Systems*, 36: 53385–53432, 2023.

Jean Kaddour, Aengus Lynch, Qi Liu, Matt J Kusner, and Ricardo Silva. Causal machine learning: A survey and open problems. *arXiv preprint arXiv:2206.15475*, 2022.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: Principles and techniques*. MIT press, 2009.

Lucas Kook, Beate Sick, and Peter Bühlmann. Distributional anchor regression. *Statistics and Computing*, 32 (3):39, 2022.

Yufan Liao, Qi Wu, and Xing Yan. Invariant random forest: Tree-based model solution for OOD generalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13772–13781. AAAI Press, 2024.

Jiashuo Liu, Tianyu Wang, Peng Cui, and Hongseok Namkoong. On the need for a language describing distribution shifts: Illustrations on tabular datasets. *Advances in Neural Information Processing Systems*, 36:51371–51408, 2023a.

Mingzhou Liu, Xiangyu Zheng, Xinwei Sun, Fang Fang, and Yizhou Wang. Which invariance should we transfer? A causal minimax learning approach. In *International Conference on Machine Learning*, volume 202, pages 22488–22527. PMLR, 2023b.

Malte Londschien, Manuel Burger, Gunnar Rätsch, and Peter Bühlmann. Domain generalization and adaptation in intensive care with anchor regression. *RSS: Data Science and Artificial Intelligence*, 2(1):udag001, 2026.

Sara Magliacane, Thijs Van Ommen, Tom Claassen, Stephan Bongers, Philip Versteeg, and Joris M Mooij. Domain adaptation by using causal inference to predict invariant conditional distributions. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Sobhan Moosavi, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. A countrywide traffic accident dataset. *arXiv preprint arXiv:1906.05409*, 2019.

Bernardino Romera Paredes, Andreas Argyriou, Nadia Berthouze, and Massimiliano Pontil. Exploiting unrelated tasks in multi-task learning. In *Artificial Intelligence and Statistics*, volume 22, pages 951–959. PMLR, 2012.

Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3:96–146, 2009.

Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: Identification and confidence intervals. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 78 (5):947–1012, 2016.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. URL `https://www.R-project.org/`.

Thomas Richardson and Peter Spirtes. Ancestral graph Markov models. *The Annals of Statistics*, 30(4):962–1030, 2002.

Mateo Rojas-Carulla, Bernhard Schölkopf, Richard Turner, and Jonas Peters. Invariant models for causal transfer learning. *Journal of Machine Learning Research*, 19(36): 1–34, 2018.

Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. The risks of invariant risk minimization. In *International Conference on Learning Representations*. PMLR, 2021. URL `https://openreview.net/forum?id=BbNIbVPJ-42`.

Dominik Rothenhäusler, Nicolai Meinshausen, Peter Bühlmann, and Jonas Peters. Anchor regression: Heterogeneous data meet causality. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 83 (2):215–246, 2021.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan R Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

Tobias Sing, Oliver Sander, Niko Beerenwinkel, and Thomas Lengauer. ROCR: Visualizing classifier performance in R. *Bioinformatics*, 21(20):3940–3941, 2005.

Beata Strack, Jonathan P DeShazo, Chris Gennings, Juan L Olmo, Sebastian Ventura, Krzysztof J Cios, and John N Clore. Impact of HbA1c measurement on hospital readmission rates: Analysis of 70,000 clinical database patient records. *BioMed Research International*, 2014(1):781670, 2014.

Terry Therneau, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning and Regression Trees*, 2026. URL `https://CRAN.R-project.org/package=rpart`. R package version 4.1.27.

Kevin Ushey, Joseph J Allaire, and Yuan Tang. *reticulate: Interface to 'Python'*, 2025. URL `https://rstudio.github.io/reticulate/`. R package version 1.44.1.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S Yu. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):8052–8072, 2022.

Marvin N. Wright and Andreas Ziegler. ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1): 1–17, 2017. doi: 10.18637/jss.v077.i01.

Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4396–4415, 2022.

Constrained Random Forest for Domain-Generalizable Classification (Supplementary Material)

Camilla Lingjærde [1] **Geir Kjetil Sandve** [2] **Arnoldo Frigessi**[3] **Sylvia Richardson**[3, 4] **Johan Pensar**[1]

[1]Department of Mathematics, University of Oslo, Oslo, Norway
[2]Department of Informatics, University of Oslo , Oslo, Norway
[3]Institute of Basic Medical Sciences, University of Oslo, Oslo, Norway
[4]MRC Biostatistics Unit , University of Cambridge , Cambridgeshire, UK

A PROOFS AND DERIVATIONS

A.1 PROOF FOR THEOREM 1

We have

$$S_G^* = \mathrm{Pa}_Y \cup \overline{\mathrm{Ch}}_Y \cup \overline{\mathrm{Sp}}_Y,$$

where

$$\overline{\mathrm{Ch}}_Y = \mathrm{Ch}_Y \setminus \{\mathrm{Ch}_Y \cap \mathrm{Ch}_Z, \mathrm{Desc}_{\mathrm{Ch}_Y \cap \mathrm{Ch}_Z}\},$$
$$\overline{\mathrm{Sp}}_Y = \mathrm{Sp}_Y \cap \mathrm{Pa}_{\overline{\mathrm{Ch}}_Y}.$$

For the sake of this proof, we rewrite

$$B \coloneqq \mathrm{Ch}_Y \cap \mathrm{Ch}_Z, \quad D \coloneqq \mathrm{Desc}(B),$$
$$C \coloneqq \mathrm{Pa}(B \cup D) \setminus \big\{\mathrm{Pa}_Y, \mathrm{Ch}_Y, \mathrm{Pa}(\mathrm{Ch}_Y \setminus \{B, D\})\big\},$$
$$S_G^* \coloneqq \mathrm{MB}_Y \setminus \big\{Z \cup B \cup C \cup D\big\},$$

which is an equivalent definition of S_G^*: we include all parents of Y, exclude all children that are $Z - Y$ colliders or descendants of those, and exclude all spouses that are only included in the Markov Blanket by virtue of being co-parents to those children.

Proof. We show that S_G^* satisfies all three conditions in Definition 1, and that uniqueness directly follows.

Invariance. First, consider a path from Z to Y that reaches Y through a parent of Y: $Z \to \cdots \rightleftharpoons X_j \to Y$, where $\rightleftharpoons$ represents an edge in either direction. Then, the parent X_j is contained by S_G^*, and conditioning on S_G^* blocks the path. Second, consider a path that reaches Y through a child of Y: $Z \to \cdots \rightleftharpoons X_j \leftarrow Y$, meaning that the path will contain at least one collider. If there is an outgoing edge from X_j towards Z, i.e. $Z \to \cdots \leftarrow X_j \leftarrow Y$, then: (i) if $X_j \in B \cup D$, then by following the direction of the edges towards Z the path will hit a collider that will block the path (since neither the collider node or any of its descendants is contained by S_G^*), (ii) if $X_j \notin B \cup D$, then X_j is contained by S_G^* and will block the path. If there is an incoming edge to X_j from the Z-side, i.e. $Z \to \cdots \rightleftharpoons X_k \to X_j \leftarrow Y$, then: (i) if $X_j \in B \cup D$, then X_j (or any of its descendants) will not be included in S_G^* and the path will be blocked, (ii) if $X_j \notin B \cup D$, then the path will be blocked by $X_k \in S_G^*$ (which must exist since $X_j \notin B$). Thus, there is no active path from Z to Y when conditioning on S_G^*, and from the soundness of d-separation we have that $Y \perp Z \mid S_G^*$.

Minimality. Let X_ℓ be any member of S_G^* and let $R = S_G^* \setminus \{X_\ell\}$. Since X_ℓ belongs to the reduced Markov blanket of Y, it is either a child, parent or spouse of Y. It can also be both a spouse and a child, or both a spouse and a parent. If X_ℓ is a child or parent of Y, there is a direct d-connecting path between X_ℓ and Y no matter what other nodes we condition on, including R. This holds regardless of whether X_ℓ simultaneously is a spouse of Y or not. Under faithfulness, this active path implies $Y \not\perp X_\ell \mid R$. If X_ℓ is only a spouse of Y, and not simultaneously a child or parent of Y, it is d-connected to Y as long as at least one common child X_j is included in the conditioning set R. Since all shared children of X_ℓ and Y are in

MB_Y, by the definition of S_G^* this always holds as long as X_ℓ and Y do not only have shared children in $\{B, D\}$. However, this is not possible since this would mean that $X_\ell \in C$ and hence excluded from S_G^*. Under faithfulness, this means that for all $X_\ell \in S_G^*$, $Y \not\perp X_\ell \mid S_G^* \setminus \{X_\ell\}$. This trivially extends to any set $V \subseteq S_G^*$, so that $Y \not\perp V \mid S_G^* \setminus V$ and thus no set of variables in S_G^* can be removed without violating Definition 1.

Sufficiency. Take any set $V \subseteq X \setminus S_G^*$. Consider first the situation where $V \subseteq \mathrm{Mb}_Y$, then we must have $V \subseteq \{B, C, D\}$. If V has at least one element in either B or D, i.e., $V \cap \{B, D\} \neq \varnothing$, then the introduction of V in the conditioning set will open a collider path between Z and Y. Under faithfulness, this means that $Y \not\perp Z \mid S_G^* \cup V$, breaking the invariance. If V has no element in B or D, i.e., $V \cap \{B, D\} = \varnothing$, then the only way for V to be in the Markov Blanket of Y is to have $V \subseteq C$ (and to be a spouse of Y). In that case, there is no active path between V and Y given S_G^*, since all the common children in B or D are excluded from the conditioning set S_G^*. The soundness of d-separation then gives $Y \perp V \mid S_G^*$, meaning V is redundant for prediction purposes.

Consider next the situation where $V \not\subseteq \mathrm{Mb}_Y$. Then we know that V is d-separated from Y when conditioning on Mb_Y. Because all of Y's parents are included in S_G^*, there can be no active path from V to Y through Y's parents. For paths through the children of Y, we know that any path through $\mathrm{Ch}_Y \not\subseteq \{B, D\}$ is closed as these children are included in S_G^*. In either case, d-separation gives us that $Y \perp V \mid S_G^*$, making V redundant for prediction purposes. For paths between V and Y through $\mathrm{Ch}_Y \in B$, we know that all collider paths between Y and parents of B are closed since $B \not\subseteq S_G^*$. All that remains are then paths through $\{B, D\}$, however, if V lies on the descendant path of Y through B or D then $V \subseteq D$ and its inclusion in the conditioning set would open the collider path between Z and Y. Under faithfulness, this would mean that $Y \not\perp Z \mid S_G^* \cup V$, breaking the invariance.

Finally, if we both have $V \cap \mathrm{Mb}_Y \neq \varnothing$ and $V \cap X \setminus \mathrm{Mb}_Y \neq \varnothing$, we can consider the subsets of V in Mb_Y and in $X \setminus \mathrm{Mb}_Y$ separately. If invariance is broken for one subset, then it is broken for all of V. Otherwise, if each subset is redundant, by d-separation, we also have $Y \perp V \mid S_G^*$. To summarize, for any $V \subseteq X \setminus S_G^*$ we have either $Y \not\perp Z \mid S_G^* \cup V$ or $Y \perp V \mid S_G^*$. Consequently, S_G^* cannot be enlarged without violating Definition 1, and is therefore the unique optimal domain-invariant set S^*.

Uniqueness. Under the assumption of faithfulness, let S be any set satisfying the three conditions in Definition 1, and decompose it as

$$S = \underbrace{(S \cap S_G^*)}_{S_{\mathrm{in}}} \cup \underbrace{(S \cap (\mathrm{MB}_Y \setminus S_G^*))}_{S_{\mathrm{MB}\setminus *}} \cup \underbrace{(S \setminus \mathrm{MB}_Y)}_{S_{\mathrm{outMB}}}.$$

We show $S_{\mathrm{MB}\setminus *} = \varnothing$, $S_{\mathrm{outMB}} = \varnothing$, and $S_{\mathrm{in}} = S_G^*$. First, by construction, $\mathrm{MB}_Y \setminus S_G^* \subseteq B \cup C \cup D$. If S contains any variable in $B \cup D$, then conditioning on it opens a $Z \to B \leftarrow Y$ collider path (or a descendant thereof), hence $Y \not\perp Z \mid X_S$, contradicting the first condition in Definition 1 under faithfulness. If instead S contains variables only from C, then these are spouses of Y whose only common children with Y lie in $B \cup D$; since $B \cup D$ cannot be in S by the previous argument, such variables are d-separated from Y given $S \cap \mathrm{MB}_Y$, and hence redundant, contradicting minimality. Therefore $S_{\mathrm{MB}\setminus *} = \varnothing$.

Next, take any $X_\ell \in S_G^* \setminus S$. Because $X_\ell \notin B \cup D$, adding X_ℓ cannot open a Z–Y path; thus invariance is preserved: $Y \perp Z \mid X_{S \cup \{\ell\}}$. Moreover, X_ℓ is a non-redundant member of the Markov blanket component retained in S_G^*, so $Y \not\perp X_\ell \mid X_S$ under faithfulness. Hence X_ℓ violates condition 3 in Definition 1 (it is neither redundant given S nor does it break invariance when added), a contradiction. Therefore $S_G^* \subseteq S$, i.e. $S_{\mathrm{in}} = S_G^*$.

Finally, consider any $X_j \in S_{\mathrm{outMB}}$. If $X_j \in D = \mathrm{Desc}(B)$, then conditioning on X_j opens a collider path of the form $Z \to B \leftarrow Y$ through a descendant of B, and by faithfulness this implies $Y \not\perp Z \mid X_S$, contradicting invariance. Hence $S \setminus \mathrm{MB}_Y$ cannot contain variables from D. If instead $X_j \notin D$, then X_j is neither in the Markov blanket of Y nor a descendant of a Z–Y collider; given that S already contains S_G^*, such variables are d-separated from Y and therefore redundant for predicting Y, contradicting minimality. Consequently, $S_{\mathrm{outMB}} = \varnothing$. The above yields $S = S_G^*$, proving uniqueness. □

A.2 PROOF FOR COROLLARY 1

Proof. Because P_{train} shares the same structural equations as P, the Y–relative independencies implied by G will hold in both distributions: $\mathcal{I}_Y(G) \subseteq \mathcal{I}_Y(P)$ and $\mathcal{I}_Y(G) \subseteq \mathcal{I}_Y(P_{\mathrm{train}})$. Under Y–faithfulness we further have that $\mathcal{I}_Y(P) = \mathcal{I}_Y(G) = \mathcal{I}_Y(P_{\mathrm{train}})$. Since $\mathcal{I}_Y$ contains precisely those independence statements needed to identify S^*, we have that $S^* = S_G^*$ is identifiable from P_{train}. □

B IMPLEMENTATION DETAILS

In this Section, we give further implementation details for the Constrained Decision Trees and Constrained Random Forests.

B.1 CONSTRAINED DECISION TREES

B.1.1 Decomposable Objective Function

Here, we give the explicit expression for the loss function evaluated at each split.

Node-wise empirical frequencies (plug-in estimates). Let t denote a node in a partially grown tree, and let $D_t = \{(x_i, y_i, z_i) : x_i \text{ falls in } t\}$ be the samples reaching t. Write

$$N_t \;=\; |D_t|, \qquad N_{y,z|t} \;=\; \sum_{(x_i,y_i,z_i)\in D_t} \mathbf{1}\{y_i = y,\; z_i = z\},$$

and define the marginals

$$N_{y|t} = \sum_{z\in\mathcal{Z}} N_{y,z|t}, \qquad N_{z|t} = \sum_{y\in\mathcal{Y}} N_{y,z|t}.$$

The empirical joint and marginal probabilities at node t are then the plug-in estimates

$$\hat{\theta}_{y,z|t} = \frac{N_{y,z|t}}{N_t}, \qquad \hat{\theta}_{y|t} = \frac{N_{y|t}}{N_t}, \qquad \hat{\theta}_{z|t} = \frac{N_{z|t}}{N_t}. \tag{8}$$

Using these, we compute the node-local plug-in functionals

$$\hat{H}(Y\,|\,t) = -\sum_{y\in\mathcal{Y}} \hat{\theta}_{y|t} \log \hat{\theta}_{y|t}, \qquad \hat{H}(Z\,|\,t) = -\sum_{z\in\mathcal{Z}} \hat{\theta}_{z|t} \log \hat{\theta}_{z|t}, \tag{9}$$

$$\hat{I}(Y;Z\,|\,t) = \sum_{y\in\mathcal{Y}} \sum_{z\in\mathcal{Z}} \hat{\theta}_{y,z|t} \log\left(\frac{\hat{\theta}_{y,z|t}}{\hat{\theta}_{y|t}\,\hat{\theta}_{z|t}}\right), \tag{10}$$

with the convention $0 \log 0 := 0$ and using the base 2 logarithm. For the whole training set D, we analogously define $\hat{H}(Z)$ from the global counts w.r.t. Z.

Leaf-decomposable objective and greedy split gain. Let $h_{\mathcal{T}}(X) \in \{1, 2, \dots M\}$ denote the leaf label induced by tree $\mathcal{T}$, and thus also the index function of the recursive partition of the input space, $\{R_m\}_{m=1}^{M}$. With a locally constant tree-based model, the constrained tree objective from (6) can be written as

$$\begin{aligned}\mathcal{L}(\mathcal{T};\lambda_1,\lambda_2) &= \hat{H}(Y \mid h_{\mathcal{T}}(X)) + \lambda_1\, \hat{I}(Y;Z \mid h_{\mathcal{T}}(X)) - \lambda_2\, \hat{H}(Z\,|\,h_{\mathcal{T}}(X)) \\ &= \sum_{m=1}^{M} P_n(h_{\mathcal{T}}(X) = m)\Big(\hat{H}(Y\,|\,m) + \lambda_1\, \hat{I}(Y;Z\,|\,m) - \lambda_2\, \hat{H}(Z\,|\,m)\Big),\end{aligned} \tag{11}$$

where $P_n(h_{\mathcal{T}}(X) = m) = N_m/N$ and $\hat{H}(\cdot\,|\,m), \hat{I}(\cdot;\cdot\,|\,m)$ are computed from the leaf contingency table via (8)–(10), as a sum of leaf-local contributions. This decomposability implies that the effect of refining a single node can be evaluated using only node-local counts.

Concretely, suppose a candidate split refines a node t into children t_L, t_R. Define the node score

$$g(t) \;=\; \hat{H}(Y\,|\,t) + \lambda_1\, \hat{I}(Y;Z\,|\,m) - \lambda_2\, \hat{H}(Z\,|\,m), \tag{12}$$

and weights

$$P_N(t) = \frac{N_t}{N},\; P_n(t_L) = \frac{N_{t_L}}{N},\; P_n(t_R) = \frac{N_{t_R}}{N}.$$

Then the change in objective (11) induced by the split is

$$\Delta\mathcal{L}_{t\to t_L,t_R} = \frac{N_{t_L}}{N} g(t_L) + \frac{N_{t_R}}{N} g(t_R) - \frac{N_t}{N} g(t), \tag{13}$$

so a greedy growth rule selects the split minimizing $\Delta\mathcal{L}$ (equivalently maximizing the constrained gain $-\Delta\mathcal{L}$). In our implementation we accept a split only if it improves the objective by at least a user-specified tolerance (an ε-improvement threshold), and we enforce standard stopping rules such as $\min$ leaf size.

B.1.2 Constraint Relaxation at Termination

In our implementation, we relax the domain-stability constraint once at the end of each root-to-leaf path. While we enforce the penalty term for all internal splits, if a node would otherwise terminate because no admissible split improves (7), we allow one final split with the λ_2 term relaxed. This (optional) computational tweak allows for the final inclusion of highly informative features, but since growth stops immediately afterwards, this is unlikely to create domain-specific sub-trees and splitting rules. The sound results in the numerical experiments in Section 5.1 in the main manuscript and Section E in the Supplementary Material suggest that this relaxation does not introduce domain dependencies; domain-specific mechanisms through $Z - Y$ colliders are not captured.

B.1.3 Pruning

We have implemented cost-complexity pruning, analogous to CART, but with the constrained objective $\mathcal{L}(\mathcal{T}; \lambda_1, \lambda_2)$ given in (11). For a complexity parameter $\alpha \geq 0$, define

$$R_\alpha(\mathcal{T}; \lambda_1, \lambda_2) \;=\; \mathcal{L}(\mathcal{T}; \lambda_1, \lambda_2) + \alpha\,|\mathcal{T}|, \tag{14}$$

where $|\mathcal{T}| = M$ is the number of leaves, which are indexed by $h_{\mathcal{T}(X)} \in \{1, \ldots, M\}$. For an internal node s, let $\mathcal{T}_s$ be the subtree rooted at s, containing $|\mathcal{T}_s|$ leaves denoted by $L(\mathcal{T}_s) \subseteq \{1, \ldots, M\}$. Let $\mathcal{T}_s^{\text{leaf}}$ denote the tree obtained by pruning $\mathcal{T}_s$ into a single leaf at s, i.e., replacing all leaves $m \in L(\mathcal{T}_s)$ by their aggregate counts at s:

$$N_{y,z|s} = \sum_{m \in L(\mathcal{T}_s)} N_{y,z|m}.$$

Using (11)-(12), the cost increase from pruning s is

$$\mathcal{L}(\mathcal{T}_s^{\text{leaf}}; \lambda_1, \lambda_2) - \mathcal{L}(\mathcal{T}_s; \lambda_1, \lambda_2) = \frac{N_s}{N} g(s) - \sum_{m \in L(\mathcal{T}_s)} \frac{N_m}{N} g(m). \tag{15}$$

We define the weakest-link value

$$\alpha_s \;=\; \frac{\mathcal{L}(\mathcal{T}_s^{\text{leaf}}; \lambda_1, \lambda_2) - \mathcal{L}(\mathcal{T}_s; \lambda_1, \lambda_2)}{|\mathcal{T}_s| - 1}, \tag{16}$$

and iteratively prune the node(s) with smallest α_s, producing a nested sequence $\mathcal{T}_0 \supset \mathcal{T}_1 \supset \cdots \supset \mathcal{T}_K$. Selection among the pruned sub-trees can be done by cross-validation using $\mathcal{L}(\mathcal{T}_k; \lambda_1, \lambda_2)$ (e.g., by a one-standard-error rule), which can be implemented efficiently due to the decomposability of $\mathcal{L}(\mathcal{T}; \lambda_1, \lambda_2)$.

B.1.4 Selecting λ_1 and λ_2

When more than two training environments are available and there is sufficient between-environment variability, we can tune (λ_1, λ_2) by grouped cross-validation that respects the environment partition (e.g., leave-one-environment-out). The objective is to select penalties that improve out-of-environment performance (e.g., test AUC/accuracy on held-out environments), optionally using a one-standard-error rule to prefer larger penalties.

In practice, however, the training domains having sufficient between-domain variability is a strong assumption. If the assumption does not hold, cross-validation can systematically favor smaller penalties for increased training data accuracy (since invariance violations may be weakly identifiable from the training split). Our empirical results suggest that choosing both penalties sufficiently large (i.e., $\lambda_1, \lambda_2 > 1$) consistently yields improvements over competing methods: once λ_1 and λ_2 are large enough to discourage environment-separating splits, performance is typically not very sensitive to further increases. We therefore recommend practical defaults such as $\lambda_1 = \lambda_2 = 10$, which demonstrates good performance for binary Y and Z. For categorical Y and Z with very different numbers of categories, however, the differences in magnitude of the penalty terms might require more careful choices of λ_1 and λ_2.

Specifically, for categorical Y and Z, we have that

$$0 \leq H(Y; Z \mid h_{\mathcal{T}}(X)) \leq \min\{H(Y \mid h_{\mathcal{T}}(X)),\, H(Z \mid h_{\mathcal{T}}(X))\} \leq \min\{\log|\mathcal{Y}|,\, \log|\mathcal{Z}|\}.$$

Moreover,

$$0 \leq I(Z; X) = H(Z) - H(Z \mid h_{\mathcal{T}}(X)) \leq H(Z) \leq \log|\mathcal{Z}|.$$

Thus, using the base 2 logarithm, the two penalty terms are bounded above by 1 when Y and Z are binary. When Z and Y has three categories, they are bounded above by 1.0986; for eight categories the upper bound is 2.0794. In this case, the penalty terms will be of larger magnitude. However, the same will hold for unpenalized part of the objective $H(Y \mid h_{\mathcal{T}}(X))$, and the relative penalization is not larger than for the binary case.

However, if the number of categories for Z is much larger or smaller than for Y, the two penalty terms will be of different magnitudes. In such a case, the choice $\lambda_1 = \lambda_2$ might not be preferred. In setting with very large $|\mathcal{Z}|$ or $|\mathcal{Y}|$, we therefore suggest to scale the penalty parameters with the penalty term magnitudes, to preserve the amount of penalization. Specifically, if $|\mathcal{Z}| >> |\mathcal{Y}|$, the second penalty term has a much larger magnitude than the first term and $H(Y \mid h_{\mathcal{T}}(X))$, and we suggest to let $\lambda_2 = \lambda_1 \log|\mathcal{Y}| / \log|\mathcal{Z}|$ and $\lambda_1 = 10$. This way, the relative penalization remains the same. Similarly, if $|\mathcal{Z}| << |\mathcal{Y}|$, then both penalty terms are of much smaller magnitude than $H(Y \mid h_{\mathcal{T}}(X))$, and we suggest to let $\lambda_2 = \lambda_1 = 10 \log|\mathcal{Y}| / \log|\mathcal{Z}|$.

B.2 CONSTRAINED RANDOM FOREST

B.2.1 Forest Predictions with Bagged Constrained Trees

A Constrained Random Forest consists of B constrained trees $\{\mathcal{T}^{(b)}\}_{b=1}^{B}$, each grown using the constrained split rule (13). For tree b, we draw a bootstrap sample $D^{(b)}$ from the training data (optionally stratified by Z to preserve environment proportions), compute node contingency tables on $D^{(b)}$, and grow $\mathcal{T}^b$ greedily by minimizing $\Delta\mathcal{L}$ subject to the same stopping rules and pruning procedure as in the single-tree case. We do not use random feature subsampling (`mtry`) in the constrained forest, since restricting the candidate set can force early splits to be chosen among variables that are suboptimal for the constraints, weakening control of domain separation and increasing variability of the penalty terms across trees.

Leaf predictions Let $h_{\mathcal{T}^{(b)}}(x) = m^{(b)}$ be the terminal leaf of x in tree b. The tree outputs the leaf-wise class probabilities

$$p_{\hat{\theta}^{(b)}}(y \mid h_{\mathcal{T}^{(b)}}(x) = m^{(b)}) \;=\; \frac{N^{(b)}_{y|m^{(b)}}}{N^{(b)}_{m^{(b)}}} \;=\; \hat{\theta}^{(b)}_{y|x}, \tag{17}$$

where $N^{(b)}_{\cdot|\cdot}$ refers to counts that are computed based on bootstrap sample b.

Forest prediction. For a new observation x, the forest aggregates tree probabilities by averaging:

$$\hat{p}_{\text{RF}}(y \mid x) \;=\; \frac{1}{B}\sum_{b=1}^{B} \hat{\theta}^{(b)}_{y|x}, \tag{18}$$

and predicts the class with maximal probability:

$$\hat{y}(x) = \arg\max_{y \in \mathcal{Y}} \big\{\hat{p}_{\text{RF}}(y \mid x)\big\}.$$

B.2.2 Optional Out-of-Bag Predictions

In the Constrained Random Forest, optional computation of out-of-bag predictions and errors is implemented; they are found by averaging only over trees for which x was not sampled into $D^{(b)}$, and can be used for model selection and diagnostics. However, since OOB is leaving out individual observations and not environments, it's a weak guide for selecting invariance penalties and/or for diagnosing OOD behavior.

B.3 COMPUTING FEATURE IMPORTANCE

For both the Constrained Decision Tree and the Constrained Random Forest, we implement feature-importance computations as optional post-processing. Specifically, users may compute (i) standard split-based importance (e.g., Gini/entropy decrease), (ii) permutation importance (measured as the drop in predictive performance after permuting a feature), and (iii) an importance score aligned with our constrained objective by aggregating the node-wise constrained gains attributable to each

split variable. In addition, we always track lightweight usage summaries: for each feature we record the total number of splits performed on that feature and an indicator of whether it was used at least once. For the forest, these counts/indicators are computed per tree and then averaged across the bagged trees, yielding the average split frequency and the average inclusion probability for each feature.

C SIMULATION STUDY DETAILS

This section provides full generative specifications for the synthetic experiments in Section 5.1 and details all evaluation summaries and tuning choices. Throughout, Z denotes the (observed) environment label, $X = (X_1, \ldots, X_p)$ the covariates, and $Y \in \{0, 1\}$ the binary target. The invariant predictive set is $X_S = \{X_2, X_3\}$, as in Figure 2.

For each SCM (models (a)–(e)) and each regime ($p = 5$ and $p = 25$), we run two evaluations. First, we compare Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) over a wide grid of penalty values $(\lambda_1, \lambda_2) \in \{0, 1, 5, 10\}^2$ against an unconstrained CART baseline (rpart; Therneau et al. [2026]) to isolate the effect of the constraints; in this evaluation we report feature *inclusion*. Second, we compare CRF to a set of invariance-oriented baselines; in this evaluation we report feature *importance* as a proxy for whether methods avoid domain separation and subsequent collider inclusion (i.e., using X_4). For Anchor Boosting we consider $\gamma \in \{1, 5, 10\}$, and for IRF we consider $\lambda \in \{1, 5, 10\}$.

Both the Constrained Decision Tree and Constrained Random Forest are implemented in R [R Core Team, 2021]. All data analysis in this paper is conducted in R, and is fully reproducible and accessible at [LINK NOT DISCLOSED DURING REVIEW]. We use Python implementations of Anchor Boosting and IRF, and access them in R via `reticulate` [Ushey et al., 2025]. For the standard CART decision tree, we use the `rpart` R package [Therneau et al., 2026]. For Random Forest, we use the `ranger` R package [Wright and Ziegler, 2017].

C.1 REPORTED METRICS

Across replicates, we report:

- **Prediction accuracy** on training and test data. For methods that output scores/probabilities, we threshold at 0.5; for tree models we use the predicted class.
- **AUC** computed from the score for the positive class, using the ROCR package [Sing et al., 2005].
- **Tree-specific invariance diagnostics** $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ on the training set (only for tree/rpart-style predictors), computed exactly as in the main text definitions using the learned partition and the training environments.
- **Feature inclusion** for tree-style comparisons (Constrained Tree / CART; and averaged over trees for Constrained Forest): an inclusion indicator is 1 if a feature appears in at least one split of the fitted tree. For the constrained forest, inclusion is the *inclusion rate* (fraction of trees in the ensemble that split on the feature at least once), and we average this rate over replicates.
- **Feature importance** for the comparative simulations of invariant methods: importance is computed method-by-method as described below and then averaged over replicates.

C.1.1 Feature Importance

In the invariance-based method comparison, we report replicate-averaged feature-importance vectors, using the following definitions:

- **Constrained Tree / Constrained Forest (CT/CRF).** We use the method's loss-based importance per feature (mean decrease in the constrained splitting objective (7) attributable to that feature). To make magnitudes comparable across methods, within each replicate we rescale to $[0, 1]$ via

$$\widetilde{I}_j = \begin{cases} \dfrac{I_j - \min_k I_k}{\max_k I_k - \min_k I_k}, & \max_k I_k > \min_k I_k, \\ I_j, & \text{otherwise}, \end{cases}$$

  and then average $\widetilde{I}$ across replicates.

- **CART (rpart).** We use rpart's variable importance (with missing entries set to 0), and apply the same within-replicate $[0, 1]$ rescaling.
- **Random Forest (RF; ranger).** We use impurity importance (mean decrease in Gini) from ranger and rescale to $[0, 1]$ within replicate.
- **ICP+rpart and ICP+RF.** We first run ICP on the training set to select a subset of features, then fit the downstream predictor on that subset. Importance is taken from the downstream predictor (rpart importance or ranger impurity importance), embedded into a length-p vector by assigning 0 to unselected features, and then rescaled to $[0, 1]$ within replicate.
- **Anchor Boosting and IRF (Python).** We use the feature-importance vector returned by the Python implementation, rescale to $[0, 1]$ within each replicate, and then average across replicates.

C.2 SCM WITH $p = 5$ FEATURES

C.2.1 Environments and Sampling

Training data are generated by sampling Z from the training domains $\{1, 2\}$ with equal probability and then sampling from $(X, Y \mid Z)$ using the corresponding SCM (that is, standard forward sampling). Test data are generated from a *single unseen* domain $Z = 3$ with shifted coefficients (specified below) and are independent of the training sample. Unless otherwise stated, exogenous noises are independent and Gaussian with common standard deviation $\sigma = 0.1$.

C.2.2 Common Components

Let $\text{logit}^{-1}(t) = 1/(1 + \exp(-t))$. Across all models, the binary response is generated as

$$(Y \mid X_2, X_3, Z) \;\sim\; \text{Bernoulli}\big(\text{logit}^{-1}\big(\beta_{y2}X_2 + \beta_{y3}X_3 + \beta_{yz}(Z)\big)\big)\,,$$

with $\beta_{y2} = \beta_{y3} = 100$ and domain-dependent offset $\beta_{yz}(Z)$ (possibly 0). The structural coefficients $\beta_{31} = 5$ and $\beta_{53} = 5$ are fixed throughout.

C.2.3 Model (a): Classical Invariance Setting (Constraint (i) Holds)

This corresponds to Figure 2(a) in the paper. The SCM is

$$\begin{aligned}
X_1 &= \varepsilon_1,\\
X_2 &= \beta_{2z}(Z) + \varepsilon_2,\\
X_3 &= \beta_{31}X_1 + \beta_{3z}(Z) + \varepsilon_3,\\
Y &\sim \text{Bernoulli}\big(\text{logit}^{-1}(\beta_{y2}X_2 + \beta_{y3}X_3)\big)\,,\\
X_4 &= \beta_{4z}(Z) + \beta_{4y}Y + \varepsilon_4, \qquad \beta_{4y} = 1,\\
X_5 &= \beta_{53}X_3 + \varepsilon_5,
\end{aligned}$$

with $(\varepsilon_1, \ldots, \varepsilon_5) \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$. Domain-specific coefficients:

$$(\beta_{2z}, \beta_{3z}, \beta_{4z}) = \begin{cases} (0.3,\ 0.3,\ -0.5), & Z = 1,\\ (-0.3,\ -0.3,\ 0.5), & Z = 2,\\ (0.5,\ -0.5,\ -10), & Z = 3 \text{ (test)}. \end{cases}$$

C.2.4 Model (b): Direct $Z \to Y$ Effect (Constraint (i) Violated; Constraint (ii) Holds)

This corresponds to Figure 2(b) in the paper. The SCM is

$$\begin{aligned}
X_1 &= \varepsilon_1, \\
X_2 &= \varepsilon_2, \\
X_3 &= \beta_{31} X_1 + \varepsilon_3, \\
Y &\sim \text{Bernoulli}\big(\text{logit}^{-1}(\beta_{y2} X_2 + \beta_{y3} X_3 + \beta_{yz}(Z))\big), \\
X_4 &= \beta_{4z}(Z) + \beta_{4y} Y + \varepsilon_4, \qquad \beta_{4y} = 1, \\
X_5 &= \beta_{5z}(Z) + \beta_{53} X_3 + \varepsilon_5,
\end{aligned}$$

with independent $\varepsilon_j \sim \mathcal{N}(0, \sigma^2)$. Domain-specific offsets:

$$(\beta_{yz}, \beta_{4z}, \beta_{5z}) = \begin{cases} (1,\ 1,\ 1), & Z = 1, \\ (-1,\ -1,\ -1), & Z = 2, \\ (1,\ 2,\ -10), & Z = 3 \text{ (test)}. \end{cases}$$

C.2.5 Model (c): Both Constraints Hold

The SCM matches model (b) but removes the direct $Z \to Y$ effect by setting $\beta_{yz}(Z) \equiv 0$:

$$Y \sim \text{Bernoulli}\big(\text{logit}^{-1}(\beta_{y2} X_2 + \beta_{y3} X_3)\big),$$

and keeps the same Z-dependent shifts in (X_4, X_5) via $(\beta_{4z}(Z), \beta_{5z}(Z))$. We use

$$(\beta_{4z}, \beta_{5z}) = \begin{cases} (1,\ 1), & Z = 1, \\ (-1,\ -1), & Z = 2, \\ (10,\ -10), & Z = 3 \text{ (test)}. \end{cases}$$

C.2.6 Model (d): Weak $Z \to Y$ Added to Model (a) (Neither Constraint Holds)

The SCM follows model (a), but adds a domain-dependent offset in the logit:

$$Y \sim \text{Bernoulli}\big(\text{logit}^{-1}(\beta_{y2} X_2 + \beta_{y3} X_3 + \beta_{yz}(Z))\big),$$

with $(X_1, X_2, X_3, X_4, X_5)$ otherwise generated as in model (a). We use

$$(\beta_{yz}, \beta_{2z}, \beta_{3z}, \beta_{4z}) = \begin{cases} (0.2,\ 0.3,\ 0.3,\ -0.5), & Z = 1, \\ (-0.2,\ -0.3,\ -0.3,\ 0.5), & Z = 2, \\ (0.5,\ 0.5,\ -0.5,\ -10), & Z = 3 \text{ (test)}. \end{cases}$$

C.2.7 Model (e): Larger Test Shift Variant of Model (a)

Training generation matches model (a) (domains $Z \in \{1, 2\}$), but the unseen test domain uses a stronger shift:

$$(\beta_{2z}, \beta_{3z}, \beta_{4z}) = (1,\ -1,\ -20) \quad \text{for } Z = 3 \text{ (test)}.$$

All other parameters follow model (a).

C.3 SCM WITH $p = 25$ FEATURES

The regime with $p = 25$ is a block-vector expansion of the SCMs with $p = 5$ features. We set $d = (d_1, \ldots, d_5) = (5, 5, 5, 5, 5)$ so that $p = \sum_{k=1}^{5} d_k = 25$. Each scalar X_k becomes a block $X_k \in \mathbb{R}^{d_k}$, and the structural equations are defined to preserve the same causal structure while keeping logits numerically stable as dimension grows.

Shared Linear Maps and Scaling

Let $\mathbf{1}_m$ denote the m-vector of ones. Define

$$A_{31} = \frac{\beta_{31}}{\sqrt{d_1}}\mathbf{1}_{d_3}\mathbf{1}_{d_1}^\top, \qquad A_{53} = \frac{\beta_{53}}{\sqrt{d_3}}\mathbf{1}_{d_5}\mathbf{1}_{d_3}^\top,$$

and weights

$$w_3 = \frac{\beta_{y3}}{\sqrt{d_3}}\mathbf{1}_{d_3}, \qquad w_2 = \frac{\beta_{y2}}{\sqrt{d_2}}\mathbf{1}_{d_2},$$

with $\beta_{31} = \beta_{53} = 5$ and $\beta_{y2} = \beta_{y3} = 100$. All noise vectors are independent, with $\varepsilon_k \sim \mathcal{N}(0, \sigma^2 I_{d_k})$ and $\sigma = 0.1$.

C.3.1 Block-Vector Generators

The block form of model (a) (and similarly (d),(e)) is:

$$\begin{aligned}
X_1 &= \varepsilon_1, \\
X_2 &= \beta_{2z}(Z)\mathbf{1}_{d_2} + \varepsilon_2, \\
X_3 &= A_{31}X_1 + \beta_{3z}(Z)\mathbf{1}_{d_3} + \varepsilon_3, \\
Y &\sim \text{Bernoulli}\big(\text{logit}^{-1}(w_3^\top X_3 + w_2^\top X_2 + \beta_{yz}(Z))\big), \\
X_4 &= \beta_{4z}(Z)\mathbf{1}_{d_4} + \beta_{4y}Y\mathbf{1}_{d_4} + \varepsilon_4, \qquad \beta_{4y} = 1, \\
X_5 &= A_{53}X_3 + \varepsilon_5,
\end{aligned}$$

while the block form of model (b) (and (c)) matches the structure with $p = 5$:

$$X_2 = \varepsilon_2, \quad X_3 = A_{31}X_1 + \varepsilon_3, \quad X_5 = \beta_{5z}(Z)\mathbf{1}_{d_5} + A_{53}X_3 + \varepsilon_5,$$

with the same Bernoulli logit for Y and the same form for X_4.

C.3.2 Domain-Specific Coefficients

Training domains remain $Z \in \{1, 2\}$ and the unseen test domain is $Z = 3$. All domain-dependent scalar coefficients $(\beta_{2z}, \beta_{3z}, \beta_{4z}, \beta_{5z}, \beta_{yz})$ are reused from the $p = 5$ setting, applied as constant shifts along the corresponding block via multiplication by $\mathbf{1}_{d_k}$. For model (e), the $p = 25$ implementation uses a slightly milder (but still large) test shift:

$$(\beta_{2z}, \beta_{3z}, \beta_{4z}) = (0.7,\ -0.7,\ -15) \quad \text{for } Z = 3 \text{ (test)}.$$

C.4 METHOD HYPERPARAMETERS AND IMPLEMENTATION CHOICES

Unless otherwise stated, all methods are trained on the pooled training sample (with observed Z where required) and evaluated on the unseen test domain.

- **Constrained Decision Tree (CTree).** We fit a single constrained tree for each penalty pair $(\lambda_1, \lambda_2) \in \{0, 1, 5, 10\}^2$ reported in the tables. The candidate split-threshold search uses $n_{\text{thresh}} = 30$, the minimum leaf size is 5, and the improvement tolerance is $\epsilon = 10^{-2}$.
- **Constrained Forest (CRF).** We fit forests with 200 trees using the same (λ_1, λ_2) grid as CT, the same minimum leaf size (5), $n_{\text{thresh}} = 30$, and improvement tolerance $\epsilon = 10^{-2}$.
- **CART (rpart).** We fit a standard classification tree using Gini splitting (default rpart settings).
- **Random Forest (RF; ranger).** We fit a ranger random forest with 200 trees, Gini splitting, and probability estimation enabled.
- **ICP+rpart / ICP+RF.** We run ICP at level $\alpha = 0.05$ on the training data using Z as the environment index and select the union of accepted sets. If ICP returns no accepted sets or fails, we fall back to using all features. To avoid numerical degeneracies in invariance testing and hence errors when running the method, we drop environments with fewer than 5 samples, drop environments with constant Y, and remove predictors that are constant within any remaining environment before calling ICP. The downstream rpart/RF is then fit on the selected subset using the same settings as above.

- **Anchor Boosting.** We use a Python implementation accessed in R via `reticulate` [Ushey et al., 2025] and evaluate $\gamma \in \{1, 5, 10\}$. The method returns predicted scores for train/test.
- **IRF.** We use a Python implementation accessed in R via `reticulate` [Ushey et al., 2025] and evaluate the invariance penalty $\lambda \in \{1, 5, 10\}$. The remaining parameters are fixed to: number of estimators 100, maximum depth 5, minimum sample periods 10, and minimum impurity decrease 0. The method returns predicted scores for train/test.

D REAL-DATA EXPERIMENT DETAILS

This section describes dataset access, domain splits, subsampling, preprocessing, and software used for the six real-data OOD classification experiments reported in Table 4. Throughout, X denotes covariates, $Y \in \{0, 1\}$ the binary label, and Z the domain (environment) label.

D.1 GENERAL EVALUATION APPROACH

For each task, we train on a small set of training environments and evaluate on a held-out environment to induce distribution shift. Because the raw datasets are large, we do not use full-sample training; instead, we construct balanced (or approximately balanced) subsamples of the training domains and a fixed-size subsample from the held-out test domain, as specified per dataset below. Table 4 reports the held-out accuracy and the corresponding standard deviation across test samples.

D.2 DATASETS AND DOMAIN SPLITS

D.2.1 ACS Income, Mobility, and Public Coverage (State Shifts)

We use three American Community Survey (ACS) tabular classification tasks based on U.S. Census microdata [Ding et al., 2021]. The target is task-specific: Income (annual income exceeds $\$50,000$), Mobility (same address as one year ago), and Public Coverage (has public health insurance), and the domain label is U.S. state ($Z =$ state). In our experiments we obtain and preprocess the ACS data using the Python whyshift package [Liu et al., 2023a], accessed from R via reticulate.[1] For all three ACS tasks, we use the year 2018 data with one held-out test state. Training and test data is sampled to have class balance, and training data is also sampled to have class-wise domain balance. We choose training states that are plausibly more similar in geography and socioeconomic context, and use a held-out test state expected to differ more substantially, in order to induce a challenging state-level distribution shift.

The domain splits and subsample sizes are:

- **ACS Income:** training states $\{\mathrm{ND}, \mathrm{SD}\}$; test state PR. We sample 600 training observations and 300 test observations. The dataset contains $p = 76$ features.
- **ACS Mobility:** training states $\{\mathrm{ND}, \mathrm{SD}\}$; test state IA. We sample 400 training observations and 400 test observations. The dataset contains $p = 63$ features.
- **ACS Public Coverage:** training states $\{\mathrm{NC}, \mathrm{SC}\}$; test state PR. We sample 400 training observations and 200 test observations. The dataset contains $p = 42$ features.

D.2.2 Diabetes Readmissions (Admission-Source Shifts)

The Diabetes Readmissions task predicts whether a patient is readmitted [Strack et al., 2014], with domains defined by admission source ($Z =$ admission source). We use a preprocessed tabular version distributed[2] and preprocessed[3] publicly, and construct training domains from Referral and Transfer admissions and a held-out test domain from Emergency admissions as suggested by Gardner et al. [2023]. We sample 1200 training observations with class and domain balance and 500 test observations from the held-out domain with class balance. The dataset contains $p = 146$ features.

[1] `https://github.com/namkoong-lab/whyshift`

[2] `https://archive.ics.uci.edu/dataset/296/diabetes+130-us+hospitals+for+years+1999-2008`

[3] `https://github.com/csinva/imodels-data/blob/master/notebooks_fetch_data/00_get_datasets_custom.ipynb`

D.2.3 US Accidents (State Shifts)

The US Accidents task predicts whether an accident is severe using a binary threshold on the reported severity [Moosavi et al., 2019], with domains defined by state ($Z = \text{state}$). We use the publicly released US Accidents dataset[4] and construct training domains from states $\{\text{NC}, \text{SC}\}$ and a held-out test domain PA. We sample 800 training observations with class and domain balance and 200 test observations from the held-out domain with class balance. The dataset contains $p = 122$ features.

D.2.4 Financial Indicators (Year Shifts)

The Financial Indicators task predicts whether a stock price increases over the next year from annual financial indicators, and generalizes across years ($Z = \text{year}$) [Carbone, 2019]. We use a public Kaggle release containing annual company financial variables and a binary label indicating whether the price increases over the subsequent year,[5] and define training environments as years 2016 and 2017 ($Z \in \{2016, 2017\}$) with a held-out test environment year 2018 ($Z = 2018$). Rows with missing values are removed prior to fitting, resulting in 755 training observations, 251 test observations, and $p = 59$ financial indicators.

D.3 PREPROCESSING AND FEATURE CONSTRUCTION

Across datasets, we apply the following preprocessing steps.

- **Missingness handling.** For the Financial and US Accidents datasets, we remove rows with missing values in the feature set. For the Financial dataset, we also drop features that are predominantly missing in the raw files (cashConversionCycle and operatingCycle in our processed version).
- **Categorical encoding.** For datasets with categorical predictors, we apply one-hot encoding using a full indicator basis (no intercept). For the Financial dataset, we one-hot encode the sector variable using this scheme. For the US Accidents dataset, we also one-hot encode weather and roadway indicator variables.
- **Derived time features (US Accidents).** From timestamps we derive hour of day, day of week, and month; these are treated as categorical and one-hot encoded.

D.4 MODEL FITTING AND SHARED SETTINGS

All ensemble methods use 200 trees. For CRF we report results for $(\lambda_1, \lambda_2) = (10, 10)$ in Table 4. For Anchor Boosting we evaluate $\gamma \in \{1, 5, 10\}$, and for IRF we evaluate $\lambda \in \{1, 5, 10\}$. Across tasks, split search uses 20 candidate thresholds per feature and a split is accepted only if it improves the objective by at least $\epsilon = 10^{-2}$. The minimum leaf size is task-specific, chosen empirically to ensure convergence within reasonable time for all tree-based methods: 30 (ACS Income), 5 (ACS Mobility), 10 (ACS Public Coverage), 5 (Diabetes), 10 (US Accidents), and 10 (Financial). CRF, ICP, Anchor Boosting and IRF are accessed through Python and R as described in Section C.

E ADDITIONAL SIMULATION RESULTS

E.1 SETTINGS WITH $p = 5$ FEATURES

Additional results for $p = 5$ features are reported in Tables 5-13. These tables provide a more granular breakdown across (λ_1, λ_2) choices and include the replicate-averaged feature inclusion summaries for the tree-based comparisons (CT/CRF versus CART), as well as the replicate-averaged feature importance vectors for the invariant-method comparisons, computed as described in Section C.1.

Overall, the supplementary tables mirror the qualitative findings in the main manuscript. First, increasing the constraint penalties tends to shift CT/CRF splitting away from environment- and collider-sensitive variables and toward the invariant predictive set $X_S = \{X_2, X_3\}$, yielding smaller train–test gaps and improved OOD accuracy/AUC in the settings where

[4] https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents

[5] https://www.kaggle.com/datasets/cnic92/200-financial-indicators-of-us-stocks-20142018

Table 5: Average results over $N = 100$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from model (a). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (a)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CTree	0	0	1.000 (0.000)	0.996 (0.004)	0.005	0.914	0.696 (0.137)	0.670 (0.147)	0.72	1.00	0.73	0.96	0.16
CTree	1	0	0.999 (0.001)	0.987 (0.006)	0.005	0.956	0.677 (0.142)	0.619 (0.154)	0.77	1.00	0.72	0.91	0.09
CTree	0	1	0.994 (0.008)	0.978 (0.015)	0.019	0.563	0.843 (0.093)	0.820 (0.105)	0.35	0.99	0.98	0.14	0.61
CTree	1	1	0.989 (0.013)	0.962 (0.027)	0.010	0.641	0.800 (0.120)	0.770 (0.138)	0.53	0.98	0.94	0.11	0.66
CTree	5	0	0.998 (0.002)	0.980 (0.014)	0.001	0.950	0.702 (0.130)	0.640 (0.157)	0.71	1.00	0.83	0.89	0.12
CTree	0	5	0.888 (0.046)	0.860 (0.055)	0.328	0.290	0.689 (0.092)	0.636 (0.106)	1.00	0.35	0.68	0.16	0.18
CTree	5	5	0.975 (0.015)	0.945 (0.022)	0.008	0.674	0.849 (0.118)	0.836 (0.135)	0.29	0.99	0.93	0.00	0.52
CTree	10	10	0.968 (0.016)	0.939 (0.020)	0.008	0.691	0.884 (0.089)	0.879 (0.094)	0.28	1.00	0.89	0.00	0.50
CRF	0	0	1.000 (0.000)	1.000 (0.001)	0.007	0.873	0.966 (0.020)	0.709 (0.111)	0.53	1.00	0.74	0.85	0.23
CRF	1	0	1.000 (0.000)	1.000 (0.001)	0.004	0.942	0.967 (0.018)	0.685 (0.145)	0.57	1.00	0.73	0.86	0.16
CRF	0	1	1.000 (0.000)	0.987 (0.005)	0.019	0.565	0.958 (0.036)	0.845 (0.040)	0.33	0.97	0.94	0.19	0.66
CRF	1	1	1.000 (0.000)	0.990 (0.007)	0.009	0.632	0.974 (0.019)	0.839 (0.062)	0.55	0.98	0.92	0.19	0.65
CRF	5	0	1.000 (0.000)	1.000 (0.001)	0.001	0.944	0.967 (0.020)	0.695 (0.141)	0.56	1.00	0.76	0.84	0.17
CRF	0	5	0.990 (0.007)	0.942 (0.038)	0.289	0.339	0.946 (0.047)	0.711 (0.087)	0.94	0.39	0.61	0.24	0.29
CRF	5	5	0.997 (0.002)	0.970 (0.011)	0.008	0.650	0.985 (0.012)	0.878 (0.053)	0.31	0.99	0.89	0.01	0.64
CRF	10	10	0.995 (0.003)	0.958 (0.012)	0.008	0.670	**0.987 (0.008)**	**0.897 (0.034)**	0.21	1.00	0.88	0.01	0.60
rpart	-	-	0.996 (0.006)	0.994 (0.009)	0.015	0.878	0.569 (0.114)	0.563 (0.114)	0.02	1.00	0.27	0.96	0.12

domain separation is most pronounced. Second, CRF typically dominates the single-tree variant for any (λ_1, λ_2) as long as they are of similar magnitude and both sufficiently large, exhibiting the same directional changes in inclusion/importance but with improved performance across replicates, consistent with variance reduction from ensembling. Finally, in the broader method comparison, the invariant-method baselines show substantially more variability across SCMs: they can perform competitively when the invariance structure is easy to recover (settings (b) and (c)), but deteriorate in the settings where highly domain-dependent features can lead to domain separation and collider inclusion (settings (a), (d), and (e)).

E.2 SETTINGS WITH $p = 25$ FEATURES

Additional results with $p = 25$ features are reported in Tables 15–23. As in the $p = 5$ setup, we report replicate-averaged feature inclusion for tree comparisons (with inclusion for CRF averaged over all trees in the ensemble and then over replicates) and replicate-averaged feature importance for the invariant-method comparisons, using the method-specific definitions in Section C.1.

The results for the settings with more features largely reflect the same patterns as the setting with fewer features, but with an amplified penalty for methods that fail to suppress the collider inclusion. CT/CRF continue to prioritize the invariant signal corresponding to the feature blocks represented by X_2 and X_3, and CRF remains the most robust overall, typically achieving the best (or near-best) OOD accuracy across SCM variants. In contrast, several baselines degrade more sharply as p increases, consistent with the enlarged opportunity set for selecting environment-specific splits and with importance profiles spreading mass across non-invariant features. Notably, in this regime unconstrained RF is less able to "accidentally" avoid problematic variables via feature subsampling, and its OOD performance correspondingly deteriorates in the settings driven by Z-dependent shifts.

Table 6: Average results over $N = 100$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from model (a). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average feature importance is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (a)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CRF	1	1	1.000 (0.000)	0.990 (0.007)	0.009	0.632	0.974 (0.019)	0.839 (0.062)	0.16	0.95	0.84	0.00	0.45
CRF	5	5	0.997 (0.002)	0.970 (0.011)	0.008	0.650	0.985 (0.012)	0.878 (0.053)	0.06	1.00	0.24	0.00	0.14
CRF	10	10	0.995 (0.003)	0.958 (0.012)	0.008	0.670	**0.987 (0.008)**	**0.897 (0.034)**	0.02	1.00	0.12	0.00	0.07
RF	-	-	1.000 (0.000)	1.000 (0.001)	-	-	0.966 (0.024)	0.750 (0.088)	0.00	0.98	0.60	0.07	0.64
ICP + rpart	-	-	0.996 (0.006)	0.994 (0.009)	0.015	0.878	0.569 (0.114)	0.563 (0.114)	0.08	0.73	0.94	0.40	0.92
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.970 (0.022)	0.757 (0.081)	0.04	0.98	0.63	0.01	0.64
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.872 (0.124)	0.545 (0.090)	0.00	0.91	0.32	0.51	0.25
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.886 (0.079)	0.539 (0.097)	0.00	0.98	0.03	0.10	0.01
AnchorBoosting	10	-	0.993 (0.075)	0.996 (0.043)	-	-	0.846 (0.141)	0.533 (0.094)	0.00	0.98	0.01	0.08	0.01
IRF	1	-	0.944 (0.010)	0.856 (0.018)	-	-	0.957 (0.031)	0.663 (0.063)	0.28	0.00	0.99	0.22	0.83
IRF	5	-	0.943 (0.010)	0.854 (0.018)	-	-	0.962 (0.029)	0.640 (0.060)	0.33	0.00	1.00	0.34	0.79
IRF	10	-	0.942 (0.010)	0.853 (0.019)	-	-	0.963 (0.027)	0.631 (0.059)	0.35	0.00	1.00	0.38	0.78

Table 7: Average results over $N = 100$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from model (b). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (b)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CTree	0	0	0.998 (0.004)	0.983 (0.014)	0.007	0.036	0.884 (0.164)	0.870 (0.154)	0.73	0.95	0.99	0.64	0.00
CTree	1	0	0.996 (0.006)	0.976 (0.015)	0.004	0.031	0.925 (0.132)	0.872 (0.159)	0.42	0.90	1.00	0.59	0.10
CTree	0	1	0.996 (0.007)	0.977 (0.013)	0.009	0.021	0.934 (0.125)	0.903 (0.129)	0.39	0.93	0.99	0.41	0.02
CTree	1	1	0.996 (0.006)	0.975 (0.015)	0.005	0.020	0.942 (0.114)	0.902 (0.128)	0.35	0.90	0.99	0.38	0.04
CTree	5	0	0.995 (0.006)	0.969 (0.018)	0.002	0.078	0.936 (0.124)	0.892 (0.134)	0.44	0.93	1.00	0.39	0.04
CTree	0	5	0.996 (0.005)	0.974 (0.015)	0.009	0.012	0.969 (0.052)	**0.937 (0.065)**	0.52	0.95	1.00	0.15	0.03
CTree	5	5	0.993 (0.007)	0.966 (0.018)	0.003	0.012	0.970 (0.048)	**0.937 (0.049)**	0.42	0.91	1.00	0.17	0.00
CTree	10	10	0.990 (0.009)	0.956 (0.021)	0.001	0.010	0.967 (0.056)	**0.925 (0.070)**	0.41	0.90	1.00	0.12	0.04
CRF	0	0	0.997 (0.002)	0.966 (0.012)	0.010	0.029	**0.990 (0.004)**	0.899 (0.113)	0.25	0.33	0.97	0.53	0.02
CRF	1	0	0.996 (0.002)	0.962 (0.013)	0.007	0.027	**0.990 (0.004)**	0.903 (0.104)	0.24	0.31	0.98	0.48	0.02
CRF	0	1	0.996 (0.002)	0.960 (0.013)	0.010	0.019	**0.990 (0.004)**	0.921 (0.072)	0.28	0.38	0.98	0.33	0.00
CRF	1	1	0.995 (0.002)	0.957 (0.013)	0.007	0.018	**0.990 (0.004)**	**0.928 (0.049)**	0.28	0.37	0.99	0.30	0.00
CRF	5	0	0.995 (0.003)	0.955 (0.012)	0.003	0.039	**0.990 (0.004)**	**0.933 (0.031)**	0.29	0.31	0.98	0.33	0.04
CRF	0	5	0.994 (0.002)	0.949 (0.012)	0.013	0.010	**0.990 (0.004)**	**0.937 (0.015)**	0.35	0.43	0.99	0.07	0.00
CRF	5	5	0.993 (0.003)	0.949 (0.012)	0.005	0.010	**0.990 (0.004)**	**0.937 (0.016)**	0.36	0.39	0.99	0.08	0.01
CRF	10	10	0.993 (0.003)	0.948 (0.012)	0.003	0.009	**0.989 (0.004)**	**0.937 (0.015)**	0.40	0.37	0.98	0.05	0.01
rpart	-	-	0.975 (0.009)	0.972 (0.008)	0.011	0.031	0.748 (0.361)	0.748 (0.217)	0.08	0.34	0.98	0.75	0.00

Table 8: Average results over $N = 100$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from model (b). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average feature importance is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (b)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CRF	1	1	0.995 (0.002)	0.957 (0.013)	0.007	0.018	0.990 (0.004)	0.928 (0.049)	0.12	0.03	1.00	0.04	0.00
CRF	5	5	0.993 (0.003)	0.949 (0.012)	0.005	0.010	0.990 (0.004)	**0.937 (0.016)**	0.15	0.03	1.00	0.02	0.00
CRF	10	10	0.993 (0.003)	0.948 (0.012)	0.003	0.009	0.989 (0.004)	**0.937 (0.015)**	0.18	0.03	1.00	0.01	0.00
RF	-	-	1.000 (0.000)	0.998 (0.002)	-	-	**0.994 (0.004)**	**0.934 (0.029)**	0.57	0.00	1.00	0.20	0.33
ICP + rpart	-	-	0.975 (0.009)	0.972 (0.008)	0.011	0.031	0.748 (0.361)	0.748 (0.217)	0.87	0.00	1.00	0.49	0.72
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	**0.993 (0.005)**	0.881 (0.120)	0.74	0.00	0.99	0.17	0.37
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.993 (0.006)	0.648 (0.167)	0.11	0.06	0.99	0.20	0.00
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	**0.993 (0.004)**	0.662 (0.168)	0.11	0.06	0.99	0.25	0.01
AnchorBoosting	10	-	1.000 (0.000)	1.000 (0.000)	-	-	**0.993 (0.005)**	0.667 (0.170)	0.08	0.05	0.86	0.43	0.04
IRF	1	-	0.994 (0.002)	0.956 (0.011)	-	-	0.992 (0.004)	**0.935 (0.020)**	0.58	0.05	1.00	0.00	0.20
IRF	5	-	0.994 (0.002)	0.957 (0.010)	-	-	0.992 (0.003)	0.930 (0.022)	0.63	0.06	1.00	0.00	0.21
IRF	10	-	0.994 (0.002)	0.957 (0.011)	-	-	0.992 (0.003)	0.928 (0.024)	0.65	0.06	1.00	0.00	0.23

Table 9: Average results over $N = 100$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from model (c). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (c)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CTree	0	0	0.998 (0.004)	0.983 (0.014)	0.007	0.031	0.909 (0.147)	0.894 (0.134)	0.66	0.95	1.00	0.48	0.02
CTree	1	0	0.996 (0.005)	0.973 (0.017)	0.004	0.032	0.928 (0.142)	0.879 (0.161)	0.31	0.96	0.99	0.46	0.09
CTree	0	1	0.996 (0.006)	0.977 (0.014)	0.009	0.019	0.929 (0.144)	0.897 (0.140)	0.46	0.97	1.00	0.29	0.01
CTree	1	1	0.996 (0.004)	0.974 (0.016)	0.005	0.016	0.941 (0.132)	0.910 (0.122)	0.35	0.98	1.00	0.19	0.03
CTree	5	0	0.995 (0.006)	0.967 (0.019)	0.002	0.065	0.941 (0.130)	0.902 (0.129)	0.35	0.94	0.99	0.33	0.06
CTree	0	5	0.996 (0.005)	0.973 (0.017)	0.008	0.011	0.978 (0.016)	0.948 (0.019)	0.50	0.98	1.00	0.10	0.01
CTree	5	5	0.994 (0.006)	0.966 (0.018)	0.002	0.012	0.978 (0.015)	0.944 (0.020)	0.42	0.94	1.00	0.15	0.03
CTree	10	10	0.991 (0.008)	0.956 (0.022)	0.001	0.007	0.978 (0.012)	0.938 (0.024)	0.45	0.89	1.00	0.02	0.01
CRF	0	0	1.000 (0.000)	0.991 (0.005)	0.010	0.040	0.995 (0.003)	0.930 (0.108)	0.53	0.86	0.99	0.51	0.04
CRF	1	0	1.000 (0.000)	0.990 (0.005)	0.005	0.042	0.995 (0.003)	0.943 (0.089)	0.45	0.84	0.99	0.47	0.09
CRF	0	1	1.000 (0.000)	0.990 (0.005)	0.011	0.027	**0.996 (0.003)**	0.954 (0.064)	0.50	0.86	0.99	0.32	0.03
CRF	1	1	1.000 (0.000)	0.990 (0.005)	0.006	0.027	**0.996 (0.002)**	0.961 (0.018)	0.47	0.85	0.99	0.30	0.05
CRF	5	0	0.999 (0.001)	0.988 (0.007)	0.002	0.072	0.995 (0.002)	**0.963 (0.012)**	0.44	0.81	0.99	0.34	0.06
CRF	0	5	1.000 (0.000)	0.989 (0.006)	0.011	0.019	**0.996 (0.002)**	**0.965 (0.012)**	0.54	0.87	0.99	0.15	0.02
CRF	5	5	0.999 (0.001)	0.986 (0.008)	0.003	0.017	**0.996 (0.002)**	**0.964 (0.012)**	0.48	0.81	0.99	0.14	0.04
CRF	10	10	0.999 (0.001)	0.983 (0.009)	0.002	0.014	0.995 (0.002)	**0.964 (0.012)**	0.50	0.79	0.99	0.09	0.04
rpart	-	-	0.975 (0.011)	0.971 (0.010)	0.012	0.028	0.831 (0.293)	0.806 (0.206)	0.12	0.41	0.96	0.68	0.00

Table 10: Average results over $N = 100$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from model (c). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average feature importance is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (c)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CRF	1	1	1.000 (0.000)	0.990 (0.005)	0.006	0.027	**0.996 (0.002)**	0.961 (0.018)	0.13	0.11	0.99	0.03	0.00
CRF	5	5	0.999 (0.001)	0.986 (0.008)	0.003	0.017	**0.996 (0.002)**	**0.964 (0.012)**	0.15	0.10	1.00	0.01	0.00
CRF	10	10	0.999 (0.001)	0.983 (0.009)	0.002	0.014	**0.995 (0.002)**	**0.964 (0.012)**	0.17	0.10	1.00	0.01	0.00
RF	-	-	1.000 (0.000)	0.998 (0.002)	-	-	0.994 (0.003)	0.934 (0.028)	0.57	0.00	1.00	0.18	0.31
ICP + rpart	-	-	0.975 (0.011)	0.971 (0.010)	0.012	0.028	0.831 (0.293)	0.806 (0.206)	0.88	0.00	0.99	0.47	0.71
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.994 (0.004)	0.882 (0.114)	0.73	0.00	0.99	0.15	0.35
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.994 (0.004)	0.641 (0.171)	0.11	0.07	0.99	0.19	0.00
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.994 (0.004)	0.663 (0.170)	0.10	0.06	0.98	0.23	0.00
AnchorBoosting	10	-	1.000 (0.000)	1.000 (0.000)	-	-	0.993 (0.006)	0.649 (0.179)	0.09	0.06	0.88	0.39	0.02
IRF	1	-	0.994 (0.002)	0.956 (0.011)	-	-	0.992 (0.003)	0.939 (0.014)	0.57	0.05	1.00	0.00	0.19
IRF	5	-	0.994 (0.002)	0.956 (0.011)	-	-	0.992 (0.003)	0.937 (0.016)	0.62	0.06	1.00	0.00	0.21
IRF	10	-	0.994 (0.002)	0.957 (0.011)	-	-	0.993 (0.003)	0.935 (0.018)	0.64	0.06	1.00	0.00	0.22

Table 11: Average results over $N = 100$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from model (d). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (d)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CTree	0	0	1.000 (0.000)	0.996 (0.004)	0.005	0.921	0.669 (0.148)	0.645 (0.152)	0.73	1.00	0.74	0.95	0.14
CTree	1	0	0.999 (0.000)	0.988 (0.006)	0.005	0.958	0.676 (0.143)	0.626 (0.157)	0.72	1.00	0.72	0.91	0.10
CTree	0	1	0.993 (0.008)	0.977 (0.016)	0.019	0.573	0.852 (0.085)	0.826 (0.102)	0.37	0.99	0.99	0.14	0.61
CTree	1	1	0.990 (0.012)	0.964 (0.026)	0.009	0.642	0.796 (0.126)	0.762 (0.145)	0.57	0.98	0.95	0.10	0.64
CTree	5	0	0.998 (0.002)	0.981 (0.013)	0.001	0.952	0.692 (0.134)	0.627 (0.155)	0.68	1.00	0.82	0.87	0.11
CTree	0	5	0.890 (0.047)	0.862 (0.056)	0.326	0.297	0.683 (0.091)	0.626 (0.108)	1.00	0.34	0.69	0.20	0.17
CTree	5	5	0.977 (0.014)	0.947 (0.021)	0.008	0.672	0.847 (0.118)	0.835 (0.128)	0.36	0.99	0.97	0.00	0.52
CTree	10	10	0.968 (0.015)	0.939 (0.020)	0.009	0.689	0.880 (0.094)	0.875 (0.097)	0.32	1.00	0.93	0.00	0.49
CRF	0	0	1.000 (0.000)	1.000 (0.001)	0.007	0.877	0.965 (0.021)	0.701 (0.117)	0.53	1.00	0.74	0.85	0.23
CRF	1	0	1.000 (0.000)	1.000 (0.001)	0.004	0.943	0.968 (0.017)	0.682 (0.147)	0.57	1.00	0.73	0.86	0.15
CRF	0	1	1.000 (0.000)	0.987 (0.006)	0.019	0.568	0.960 (0.035)	0.843 (0.041)	0.33	0.97	0.94	0.18	0.66
CRF	1	1	1.000 (0.000)	0.991 (0.007)	0.009	0.634	0.974 (0.018)	0.840 (0.060)	0.56	0.98	0.92	0.19	0.65
CRF	5	0	1.000 (0.000)	1.000 (0.001)	0.001	0.945	0.966 (0.021)	0.690 (0.145)	0.56	1.00	0.76	0.84	0.17
CRF	0	5	0.991 (0.007)	0.944 (0.039)	0.289	0.343	0.946 (0.046)	0.708 (0.088)	0.94	0.40	0.61	0.25	0.29
CRF	5	5	0.997 (0.002)	0.970 (0.011)	0.008	0.652	0.986 (0.011)	0.877 (0.049)	0.31	0.99	0.89	0.01	0.63
CRF	10	10	0.995 (0.003)	0.959 (0.013)	0.008	0.673	**0.987 (0.005)**	**0.895 (0.034)**	0.21	1.00	0.88	0.01	0.59
rpart	-	-	0.997 (0.005)	0.995 (0.008)	0.012	0.890	0.563 (0.112)	0.551 (0.111)	0.02	1.00	0.25	0.96	0.11

Table 12: Average results over $N = 100$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from model (d). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average feature importance is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (d)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CRF	1	1	1.000 (0.000)	0.991 (0.007)	0.009	0.634	0.974 (0.018)	0.840 (0.060)	0.16	0.94	0.84	0.00	0.44
CRF	5	5	0.997 (0.002)	0.970 (0.011)	0.008	0.652	0.986 (0.011)	0.877 (0.049)	0.06	1.00	0.24	0.00	0.13
CRF	10	10	0.995 (0.003)	0.959 (0.013)	0.008	0.673	**0.987 (0.005)**	**0.895 (0.034)**	0.02	1.00	0.12	0.00	0.06
RF	-	-	1.000 (0.000)	1.000 (0.001)	-	-	0.964 (0.027)	0.750 (0.088)	0.00	0.99	0.59	0.07	0.62
ICP + rpart	-	-	0.997 (0.005)	0.995 (0.008)	0.012	0.890	0.563 (0.112)	0.551 (0.111)	0.07	0.75	0.94	0.42	0.92
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.970 (0.020)	0.758 (0.076)	0.04	0.98	0.62	0.01	0.63
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.866 (0.131)	0.538 (0.085)	0.01	0.92	0.29	0.52	0.23
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.869 (0.117)	0.531 (0.091)	0.00	0.97	0.02	0.11	0.01
AnchorBoosting	10	-	0.993 (0.075)	0.996 (0.043)	-	-	0.856 (0.131)	0.529 (0.094)	0.00	0.99	0.01	0.06	0.00
IRF	1	-	0.944 (0.010)	0.856 (0.019)	-	-	0.958 (0.031)	0.659 (0.064)	0.28	0.00	0.99	0.22	0.83
IRF	5	-	0.943 (0.010)	0.853 (0.018)	-	-	0.962 (0.029)	0.636 (0.062)	0.33	0.00	0.99	0.33	0.81
IRF	10	-	0.942 (0.010)	0.852 (0.018)	-	-	0.963 (0.028)	0.628 (0.060)	0.35	0.00	0.99	0.37	0.80

Table 13: Average results over $N = 100$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from model (e). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (e)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CTree	0	0	1.000 (0.000)	0.996 (0.004)	0.005	0.914	0.595 (0.110)	0.571 (0.109)	0.72	1.00	0.73	0.96	0.16
CTree	1	0	0.999 (0.001)	0.987 (0.006)	0.005	0.956	0.582 (0.063)	0.537 (0.063)	0.77	1.00	0.72	0.91	0.09
CTree	0	1	0.994 (0.008)	0.978 (0.015)	0.019	0.563	0.596 (0.050)	0.583 (0.041)	0.35	0.99	0.98	0.14	0.61
CTree	1	1	0.989 (0.013)	0.962 (0.027)	0.010	0.641	0.579 (0.042)	0.570 (0.044)	0.53	0.98	0.94	0.11	0.66
CTree	5	0	0.998 (0.002)	0.980 (0.014)	0.001	0.950	0.579 (0.058)	0.542 (0.057)	0.71	1.00	0.83	0.89	0.12
CTree	0	5	0.888 (0.046)	0.860 (0.055)	0.328	0.290	0.611 (0.067)	0.550 (0.072)	1.00	0.35	0.68	0.16	0.18
CTree	5	5	0.975 (0.015)	0.945 (0.022)	0.008	0.674	0.606 (0.062)	0.599 (0.060)	0.29	0.99	0.93	0.00	0.52
CTree	10	10	0.968 (0.016)	0.939 (0.020)	0.008	0.691	0.633 (0.065)	**0.632 (0.065)**	0.28	1.00	0.89	0.00	0.50
CRF	0	0	1.000 (0.000)	1.000 (0.001)	0.007	0.873	0.798 (0.066)	0.542 (0.040)	0.53	1.00	0.74	0.85	0.23
CRF	1	0	1.000 (0.000)	1.000 (0.001)	0.004	0.942	0.801 (0.055)	0.540 (0.045)	0.57	1.00	0.73	0.86	0.16
CRF	0	1	1.000 (0.000)	0.987 (0.005)	0.019	0.565	0.731 (0.133)	0.584 (0.028)	0.33	0.97	0.94	0.19	0.66
CRF	1	1	1.000 (0.000)	0.990 (0.007)	0.009	0.632	0.795 (0.079)	0.584 (0.036)	0.55	0.98	0.92	0.19	0.65
CRF	5	0	1.000 (0.000)	1.000 (0.001)	0.001	0.944	0.792 (0.076)	0.542 (0.044)	0.56	1.00	0.76	0.84	0.17
CRF	0	5	0.990 (0.007)	0.942 (0.038)	0.289	0.339	0.826 (0.130)	0.590 (0.055)	0.94	0.39	0.61	0.24	0.29
CRF	5	5	0.997 (0.002)	0.970 (0.011)	0.008	0.650	0.833 (0.100)	0.604 (0.041)	0.31	0.99	0.89	0.01	0.64
CRF	10	10	0.995 (0.003)	0.958 (0.012)	0.008	0.670	**0.852 (0.094)**	**0.620 (0.045)**	0.21	1.00	0.88	0.01	0.60
rpart	-	-	0.996 (0.006)	0.994 (0.009)	0.015	0.878	0.515 (0.031)	0.511 (0.038)	0.02	1.00	0.27	0.96	0.12

Table 14: Average results over $N = 100$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from model (e). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average feature importance is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (e)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	X_1	X_2	X_3	X_4	X_5
CRF	1	1	1.000 (0.000)	0.990 (0.007)	0.009	0.632	0.795 (0.079)	0.584 (0.036)	0.16	0.95	0.84	0.00	0.45
CRF	5	5	0.997 (0.002)	0.970 (0.011)	0.008	0.650	0.833 (0.100)	0.604 (0.041)	0.06	1.00	0.24	0.00	0.14
CRF	10	10	0.995 (0.003)	0.958 (0.012)	0.008	0.670	**0.852 (0.094)**	**0.620 (0.045)**	0.02	1.00	0.12	0.00	0.07
RF	-	-	1.000 (0.000)	1.000 (0.001)	-	-	0.756 (0.155)	0.552 (0.038)	0.00	0.98	0.60	0.07	0.64
ICP + rpart	-	-	0.996 (0.006)	0.994 (0.009)	0.015	0.878	0.515 (0.031)	0.511 (0.038)	0.08	0.73	0.94	0.40	0.92
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.788 (0.150)	0.553 (0.037)	0.04	0.98	0.63	0.01	0.64
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.650 (0.132)	0.506 (0.036)	0.00	0.91	0.32	0.51	0.25
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.657 (0.147)	0.506 (0.035)	0.00	0.98	0.03	0.10	0.01
AnchorBoosting	10	-	0.993 (0.075)	0.996 (0.043)	-	-	0.626 (0.161)	0.504 (0.035)	0.00	0.98	0.01	0.08	0.01
IRF	1	-	0.944 (0.010)	0.856 (0.018)	-	-	0.768 (0.111)	0.527 (0.029)	0.28	0.00	0.99	0.22	0.83
IRF	5	-	0.943 (0.010)	0.854 (0.018)	-	-	0.810 (0.108)	0.521 (0.029)	0.33	0.00	1.00	0.34	0.79
IRF	10	-	0.942 (0.010)	0.853 (0.019)	-	-	0.824 (0.106)	0.519 (0.028)	0.35	0.00	1.00	0.38	0.78

Table 15: Average results over $N = 10$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from the scaled-up version of model (a). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature vector, averaged over each vector entry, is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (a)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CTree	0	0	1.000 (0.000)	0.998 (0.001)	0.002	0.941	0.744 (0.099)	0.734 (0.104)	0.18	0.36	0.32	0.24	0.14
CTree	1	0	1.000 (0.000)	0.995 (0.003)	0.002	0.972	0.741 (0.110)	0.669 (0.165)	0.16	0.34	0.28	0.28	0.14
CTree	0	1	0.996 (0.002)	0.988 (0.003)	0.026	0.555	0.862 (0.021)	0.856 (0.023)	0.04	0.28	0.14	0.06	0.36
CTree	1	1	0.986 (0.015)	0.962 (0.030)	0.010	0.695	0.807 (0.143)	0.799 (0.151)	0.06	0.32	0.20	0.02	0.34
CTree	5	0	1.000 (0.000)	0.993 (0.003)	0.001	0.969	0.762 (0.126)	0.721 (0.171)	0.14	0.30	0.30	0.22	0.14
CTree	0	5	0.945 (0.030)	0.890 (0.041)	0.287	0.364	0.780 (0.057)	0.733 (0.068)	0.66	0.30	0.34	0.24	0.40
CTree	5	5	0.971 (0.016)	0.943 (0.023)	0.009	0.714	0.728 (0.201)	0.719 (0.208)	0.00	0.30	0.28	0.00	0.10
CTree	10	10	0.970 (0.012)	0.944 (0.013)	0.009	0.708	0.848 (0.172)	0.836 (0.168)	0.00	0.24	0.34	0.00	0.06
CRF	0	0	1.000 (0.000)	1.000 (0.000)	0.003	0.931	0.959 (0.011)	0.705 (0.098)	0.16	0.36	0.20	0.29	0.13
CRF	1	0	1.000 (0.000)	1.000 (0.000)	0.002	0.965	0.959 (0.009)	0.729 (0.097)	0.16	0.33	0.23	0.28	0.15
CRF	0	1	1.000 (0.000)	0.993 (0.002)	0.017	0.567	0.948 (0.019)	0.845 (0.019)	0.07	0.26	0.15	0.03	0.39
CRF	1	1	1.000 (0.000)	0.993 (0.005)	0.007	0.653	0.976 (0.013)	0.852 (0.017)	0.07	0.36	0.27	0.04	0.25
CRF	5	0	1.000 (0.000)	1.000 (0.000)	0.001	0.961	0.965 (0.009)	0.739 (0.055)	0.15	0.31	0.27	0.27	0.12
CRF	0	5	1.000 (0.000)	0.997 (0.002)	0.255	0.417	0.963 (0.017)	0.812 (0.037)	0.22	0.29	0.25	0.24	0.22
CRF	5	5	0.998 (0.001)	0.973 (0.005)	0.007	0.682	**0.993 (0.005)**	0.902 (0.026)	0.03	0.28	0.33	0.00	0.12
CRF	10	10	0.996 (0.002)	0.963 (0.006)	0.006	0.695	**0.994 (0.005)**	**0.927 (0.029)**	0.03	0.24	0.32	0.00	0.10
rpart	-	-	0.999 (0.001)	0.999 (0.002)	0.007	0.941	0.543 (0.071)	0.535 (0.076)	0.00	0.24	0.04	0.20	0.02

Table 16: Average results over $N = 10$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from the scaled-up version of model (a). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average importance of each feature vector, averaged over each vector entry, is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (a)

			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CRF	1	1	1.000 (0.000)	0.993 (0.005)	0.007	0.653	0.976 (0.013)	0.852 (0.017)	0.00	0.50	0.24	0.01	0.28
CRF	5	5	0.998 (0.001)	0.973 (0.005)	0.007	0.682	**0.993 (0.005)**	0.902 (0.026)	0.00	0.56	0.13	0.00	0.04
CRF	10	10	0.996 (0.002)	0.963 (0.006)	0.006	0.695	**0.994 (0.005)**	**0.927 (0.029)**	0.00	0.50	0.07	0.00	0.02
RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.923 (0.012)	0.595 (0.101)	0.00	0.77	0.24	0.34	0.41
ICP + rpart	-	-	1.000 (0.001)	0.999 (0.001)	0.003	0.971	0.523 (0.072)	0.516 (0.073)	0.00	0.77	0.02	0.44	0.00
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.547 (0.104)	0.492 (0.025)	0.00	0.63	0.02	0.37	0.00
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.546 (0.234)	0.534 (0.074)	0.00	0.19	0.04	0.13	0.02
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.708 (0.213)	0.528 (0.115)	0.00	0.22	0.00	0.05	0.00
AnchorBoosting	10	-	1.000 (0.000)	1.000 (0.000)	-	-	0.815 (0.176)	0.514 (0.054)	0.00	0.17	0.00	0.07	0.00
IRF	1	-	0.958 (0.009)	0.837 (0.013)	-	-	**0.992 (0.005)**	0.742 (0.030)	0.03	0.00	0.65	0.03	0.74
IRF	5	-	0.962 (0.007)	0.836 (0.013)	-	-	0.989 (0.005)	0.721 (0.035)	0.05	0.00	0.74	0.14	0.71
IRF	10	-	0.962 (0.007)	0.837 (0.012)	-	-	0.987 (0.008)	0.686 (0.068)	0.07	0.00	0.77	0.21	0.68

Table 17: Average results over $N = 10$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from the scaled-up version of model (b). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature vector, averaged over each vector entry, is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (b)

			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CTree	0	0	0.997 (0.001)	0.975 (0.006)	0.023	0.055	0.951 (0.016)	0.936 (0.006)	0.08	0.08	0.56	0.20	0.16
CTree	1	0	0.996 (0.001)	0.970 (0.007)	0.007	0.035	0.952 (0.020)	0.935 (0.016)	0.06	0.06	0.72	0.14	0.10
CTree	0	1	0.997 (0.001)	0.973 (0.005)	0.013	0.035	0.922 (0.129)	0.893 (0.142)	0.14	0.06	0.66	0.18	0.14
CTree	1	1	0.996 (0.001)	0.971 (0.006)	0.005	0.019	0.965 (0.022)	0.940 (0.019)	0.08	0.04	0.74	0.16	0.00
CTree	5	0	0.995 (0.001)	0.963 (0.004)	0.001	0.047	0.984 (0.008)	0.947 (0.011)	0.04	0.04	0.82	0.04	0.04
CTree	0	5	0.997 (0.001)	0.972 (0.007)	0.007	0.013	0.965 (0.031)	0.941 (0.021)	0.12	0.10	0.82	0.12	0.00
CTree	5	5	0.995 (0.001)	0.964 (0.006)	0.001	0.007	0.982 (0.012)	0.946 (0.011)	0.06	0.06	0.82	0.06	0.00
CTree	10	10	0.995 (0.002)	0.958 (0.012)	0.001	0.003	0.985 (0.006)	0.940 (0.010)	0.08	0.08	0.82	0.00	0.00
CRF	0	0	1.000 (0.000)	0.992 (0.003)	0.017	0.044	**0.996 (0.001)**	**0.964 (0.008)**	0.15	0.10	0.63	0.18	0.09
CRF	1	0	1.000 (0.000)	0.991 (0.002)	0.004	0.029	**0.996 (0.002)**	**0.965 (0.007)**	0.12	0.11	0.69	0.13	0.05
CRF	0	1	1.000 (0.000)	0.990 (0.002)	0.012	0.028	**0.996 (0.002)**	**0.964 (0.010)**	0.12	0.12	0.67	0.12	0.07
CRF	1	1	1.000 (0.000)	0.990 (0.002)	0.004	0.019	**0.996 (0.002)**	**0.964 (0.008)**	0.10	0.12	0.72	0.09	0.03
CRF	5	0	0.999 (0.000)	0.986 (0.003)	0.001	0.025	**0.996 (0.002)**	**0.966 (0.008)**	0.08	0.10	0.72	0.06	0.04
CRF	0	5	1.000 (0.000)	0.990 (0.003)	0.008	0.013	**0.996 (0.002)**	**0.963 (0.008)**	0.11	0.12	0.74	0.07	0.02
CRF	5	5	0.999 (0.000)	0.986 (0.003)	0.001	0.009	**0.997 (0.002)**	**0.965 (0.007)**	0.07	0.10	0.74	0.04	0.01
CRF	10	10	0.999 (0.000)	0.984 (0.004)	0.001	0.006	**0.997 (0.002)**	**0.966 (0.008)**	0.06	0.09	0.74	0.02	0.01
rpart	-	-	0.979 (0.004)	0.977 (0.003)	0.014	0.026	0.295 (0.375)	0.592 (0.188)	0.00	0.00	0.32	0.16	0.04

Table 18: Average results over $N = 10$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from the scaled-up version of model (b). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average importance of each feature vector, averaged over each vector entry, is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (b)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CRF	1	1	1.000 (0.000)	0.990 (0.002)	0.004	0.019	**0.996 (0.002)**	**0.964 (0.008)**	0.00	0.00	0.54	0.01	0.03
CRF	5	5	0.999 (0.000)	0.986 (0.003)	0.001	0.009	**0.997 (0.002)**	**0.965 (0.007)**	0.00	0.00	0.57	0.00	0.00
CRF	10	10	0.999 (0.000)	0.984 (0.004)	0.001	0.006	**0.997 (0.002)**	**0.966 (0.008)**	0.00	0.00	0.56	0.00	0.00
RF	-	-	1.000 (0.000)	1.000 (0.001)	-	-	0.909 (0.177)	0.682 (0.222)	0.00	0.01	0.67	0.12	0.48
ICP + rpart	-	-	0.960 (0.045)	0.950 (0.050)	0.007	0.013	0.937 (0.063)	0.923 (0.066)	0.07	0.02	0.46	0.01	0.07
ICP + RF	-	-	1.000 (0.000)	1.000 (0.001)	-	-	0.976 (0.060)	0.925 (0.067)	0.05	0.05	0.42	0.01	0.07
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.994 (0.003)	0.509 (0.031)	0.00	0.00	0.42	0.06	0.06
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.993 (0.004)	0.511 (0.033)	0.00	0.00	0.40	0.06	0.08
AnchorBoosting	10	-	1.000 (0.000)	1.000 (0.000)	-	-	0.989 (0.014)	0.544 (0.132)	0.00	0.00	0.27	0.07	0.13
IRF	1	-	0.998 (0.001)	0.977 (0.005)	-	-	**0.997 (0.001)**	0.955 (0.008)	0.02	0.00	0.77	0.00	0.10
IRF	5	-	0.998 (0.001)	0.975 (0.004)	-	-	**0.997 (0.001)**	0.958 (0.011)	0.02	0.00	0.77	0.00	0.04
IRF	10	-	0.998 (0.001)	0.976 (0.004)	-	-	**0.997 (0.001)**	0.956 (0.014)	0.03	0.00	0.79	0.00	0.04

Table 19: Average results over $N = 10$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from the scaled-up version of model (c). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature vector, averaged over each vector entry, is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (c)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CTree	0	0	0.997 (0.001)	0.975 (0.007)	0.018	0.032	0.938 (0.044)	0.919 (0.029)	0.10	0.10	0.70	0.16	0.02
CTree	1	0	0.997 (0.001)	0.971 (0.007)	0.004	0.019	0.973 (0.027)	0.942 (0.017)	0.06	0.14	0.86	0.12	0.00
CTree	0	1	0.997 (0.001)	0.975 (0.004)	0.013	0.024	0.949 (0.042)	0.924 (0.027)	0.06	0.12	0.74	0.16	0.00
CTree	1	1	0.997 (0.001)	0.971 (0.007)	0.004	0.014	0.972 (0.028)	0.941 (0.018)	0.06	0.12	0.84	0.10	0.00
CTree	5	0	0.995 (0.002)	0.965 (0.009)	0.000	0.009	0.978 (0.021)	0.946 (0.015)	0.02	0.04	0.88	0.02	0.02
CTree	0	5	0.997 (0.001)	0.974 (0.006)	0.008	0.015	0.963 (0.040)	0.933 (0.025)	0.02	0.12	0.76	0.14	0.00
CTree	5	5	0.996 (0.002)	0.965 (0.008)	0.001	0.005	0.983 (0.009)	0.946 (0.010)	0.06	0.10	0.88	0.02	0.00
CTree	10	10	0.995 (0.002)	0.960 (0.013)	0.000	0.004	0.984 (0.007)	0.947 (0.012)	0.02	0.10	0.90	0.00	0.00
CRF	0	0	1.000 (0.000)	0.991 (0.004)	0.015	0.035	**0.997 (0.002)**	0.966 (0.009)	0.16	0.11	0.67	0.16	0.05
CRF	1	0	1.000 (0.000)	0.989 (0.004)	0.004	0.025	**0.997 (0.001)**	**0.968 (0.006)**	0.12	0.11	0.73	0.12	0.03
CRF	0	1	1.000 (0.000)	0.989 (0.004)	0.011	0.023	**0.997 (0.001)**	0.965 (0.008)	0.12	0.12	0.71	0.12	0.03
CRF	1	1	1.000 (0.000)	0.988 (0.003)	0.004	0.017	**0.997 (0.001)**	0.967 (0.008)	0.09	0.12	0.75	0.08	0.02
CRF	5	0	0.999 (0.000)	0.985 (0.004)	0.001	0.029	**0.997 (0.002)**	**0.969 (0.006)**	0.08	0.09	0.74	0.05	0.02
CRF	0	5	1.000 (0.000)	0.989 (0.003)	0.009	0.012	**0.997 (0.001)**	0.966 (0.006)	0.11	0.13	0.76	0.07	0.01
CRF	5	5	0.999 (0.000)	0.984 (0.004)	0.001	0.008	**0.997 (0.002)**	**0.970 (0.006)**	0.06	0.10	0.75	0.03	0.01
CRF	10	10	0.999 (0.000)	0.983 (0.004)	0.001	0.005	**0.997 (0.002)**	**0.971 (0.005)**	0.05	0.08	0.75	0.01	0.00
rpart	-	-	0.981 (0.006)	0.977 (0.005)	0.014	0.019	0.501 (0.470)	0.724 (0.230)	0.00	0.00	0.44	0.16	0.00

Table 20: Average results over $N = 10$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from the scaled-up version of model (c). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average importance of each feature vector, averaged over each vector entry, is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (c)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CRF	1	1	1.000 (0.000)	0.988 (0.003)	0.004	0.017	**0.997** (0.001)	0.967 (0.008)	0.00	0.00	0.48	0.01	0.01
CRF	5	5	0.999 (0.000)	0.984 (0.004)	0.001	0.008	**0.997** (0.002)	**0.970 (0.006)**	0.00	0.00	0.48	0.00	0.00
CRF	10	10	0.999 (0.000)	0.983 (0.004)	0.001	0.005	**0.997** (0.002)	**0.971 (0.005)**	0.00	0.00	0.50	0.00	0.00
RF	-	-	1.000 (0.000)	1.000 (0.001)	-	-	0.953 (0.092)	0.688 (0.234)	0.00	0.01	0.73	0.13	0.42
ICP + rpart	-	-	0.957 (0.049)	0.948 (0.060)	0.006	0.006	0.851 (0.280)	0.881 (0.152)	0.07	0.02	0.48	0.01	0.07
ICP + RF	-	-	1.000 (0.000)	1.000 (0.001)	-	-	0.976 (0.057)	0.923 (0.070)	0.05	0.04	0.45	0.01	0.05
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.986 (0.024)	0.508 (0.025)	0.00	0.00	0.41	0.05	0.01
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.984 (0.034)	0.508 (0.025)	0.00	0.00	0.42	0.06	0.05
AnchorBoosting	10	-	1.000 (0.000)	1.000 (0.000)	-	-	0.985 (0.030)	0.508 (0.025)	0.00	0.00	0.41	0.06	0.06
IRF	1	-	0.998 (0.000)	0.977 (0.005)	-	-	**0.997 (0.001)**	0.963 (0.010)	0.02	0.00	0.76	0.00	0.07
IRF	5	-	0.998 (0.000)	0.976 (0.004)	-	-	**0.998 (0.001)**	0.967 (0.006)	0.02	0.00	0.75	0.00	0.03
IRF	10	-	0.998 (0.000)	0.975 (0.004)	-	-	**0.998 (0.001)**	0.966 (0.011)	0.03	0.00	0.77	0.00	0.03

Table 21: Average results over $N = 10$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from the scaled-up version of model (d). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature vector, averaged over each vector entry, is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (d)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CTree	0	0	1.000 (0.000)	0.998 (0.001)	0.001	0.923	0.732 (0.114)	0.724 (0.108)	0.16	0.36	0.32	0.24	0.12
CTree	1	0	1.000 (0.000)	0.995 (0.003)	0.002	0.972	0.740 (0.110)	0.671 (0.167)	0.14	0.32	0.28	0.28	0.14
CTree	0	1	0.996 (0.002)	0.988 (0.004)	0.026	0.555	0.861 (0.024)	0.856 (0.026)	0.04	0.28	0.16	0.04	0.36
CTree	1	1	0.986 (0.015)	0.963 (0.031)	0.010	0.693	0.802 (0.141)	0.793 (0.150)	0.06	0.30	0.20	0.02	0.32
CTree	5	0	1.000 (0.000)	0.994 (0.003)	0.001	0.969	0.764 (0.129)	0.711 (0.166)	0.18	0.30	0.30	0.22	0.14
CTree	0	5	0.941 (0.030)	0.887 (0.042)	0.298	0.348	0.790 (0.052)	0.730 (0.066)	0.60	0.34	0.34	0.22	0.40
CTree	5	5	0.971 (0.016)	0.942 (0.022)	0.009	0.713	0.725 (0.198)	0.714 (0.206)	0.00	0.30	0.28	0.00	0.10
CTree	10	10	0.970 (0.012)	0.944 (0.013)	0.009	0.708	0.847 (0.171)	0.835 (0.169)	0.00	0.24	0.34	0.00	0.06
CRF	0	0	1.000 (0.000)	1.000 (0.000)	0.004	0.927	0.957 (0.012)	0.713 (0.103)	0.16	0.36	0.20	0.28	0.13
CRF	1	0	1.000 (0.000)	1.000 (0.000)	0.002	0.965	0.957 (0.010)	0.725 (0.101)	0.16	0.33	0.23	0.28	0.15
CRF	0	1	1.000 (0.000)	0.994 (0.002)	0.017	0.567	0.945 (0.028)	0.845 (0.024)	0.07	0.26	0.14	0.03	0.38
CRF	1	1	1.000 (0.000)	0.993 (0.005)	0.007	0.652	0.972 (0.014)	0.852 (0.021)	0.07	0.36	0.27	0.04	0.25
CRF	5	0	1.000 (0.000)	1.000 (0.000)	0.001	0.961	0.964 (0.009)	0.740 (0.059)	0.15	0.31	0.27	0.27	0.12
CRF	0	5	1.000 (0.000)	0.997 (0.002)	0.256	0.416	0.960 (0.018)	0.814 (0.040)	0.22	0.29	0.25	0.24	0.22
CRF	5	5	0.998 (0.001)	0.974 (0.005)	0.007	0.682	**0.992 (0.008)**	0.900 (0.029)	0.04	0.28	0.33	0.00	0.12
CRF	10	10	0.996 (0.002)	0.963 (0.006)	0.006	0.695	**0.994 (0.004)**	**0.925 (0.031)**	0.03	0.24	0.32	0.00	0.10
rpart	-	-	1.000 (0.001)	0.999 (0.001)	0.004	0.932	0.557 (0.104)	0.552 (0.107)	0.00	0.24	0.02	0.20	0.06

Table 22: Average results over $N = 10$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from the scaled-up version of model (d). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average importance of each feature vector, averaged over each vector entry, is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (d)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CRF	1	1	1.000 (0.000)	0.993 (0.005)	0.007	0.652	0.972 (0.014)	0.852 (0.021)	0.00	0.50	0.24	0.01	0.28
CRF	5	5	0.998 (0.001)	0.974 (0.005)	0.007	0.682	**0.992 (0.008)**	0.900 (0.029)	0.00	0.55	0.13	0.00	0.04
CRF	10	10	0.996 (0.002)	0.963 (0.006)	0.006	0.695	**0.994 (0.004)**	**0.925 (0.031)**	0.00	0.51	0.07	0.00	0.02
RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.922 (0.024)	0.606 (0.114)	0.00	0.77	0.25	0.34	0.41
ICP + rpart	-	-	1.000 (0.000)	1.000 (0.001)	0.002	0.982	0.523 (0.072)	0.518 (0.072)	0.00	0.77	0.02	0.41	0.00
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.546 (0.109)	0.494 (0.028)	0.00	0.62	0.02	0.36	0.00
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.539 (0.223)	0.537 (0.072)	0.00	0.19	0.04	0.13	0.03
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.709 (0.209)	0.530 (0.115)	0.00	0.22	0.00	0.05	0.00
AnchorBoosting	10	-	1.000 (0.000)	1.000 (0.000)	-	-	0.763 (0.218)	0.517 (0.056)	0.00	0.17	0.00	0.06	0.00
IRF	1	-	0.958 (0.009)	0.838 (0.015)	-	-	0.991 (0.005)	0.744 (0.033)	0.03	0.00	0.66	0.03	0.75
IRF	5	-	0.962 (0.007)	0.837 (0.014)	-	-	0.988 (0.007)	0.724 (0.038)	0.05	0.00	0.74	0.14	0.72
IRF	10	-	0.963 (0.008)	0.837 (0.013)	-	-	0.985 (0.010)	0.689 (0.072)	0.07	0.00	0.76	0.21	0.68

Table 23: Average results over $N = 10$ replicates, for our Constrained Decision Tree (CTree) and Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as the standard rpart, applied to data simulated from the scaled-up version of model (e). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average inclusion indicator of each feature vector, averaged over each vector entry, is shown as well. The highest values of the test data AUC and prediction accuracy, as well as all estimates within two standard errors of these, are marked in bold.

Model (e)													
			Training				Test		Inclusion				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CTree	0	0	1.000 (0.000)	0.998 (0.001)	0.002	0.941	0.655 (0.077)	0.642 (0.082)	0.18	0.36	0.32	0.24	0.14
CTree	1	0	1.000 (0.000)	0.995 (0.003)	0.002	0.972	0.660 (0.074)	0.596 (0.106)	0.16	0.34	0.28	0.28	0.14
CTree	0	1	0.996 (0.002)	0.988 (0.003)	0.026	0.555	0.727 (0.025)	0.719 (0.031)	0.04	0.28	0.14	0.06	0.36
CTree	1	1	0.986 (0.015)	0.962 (0.030)	0.010	0.695	0.689 (0.096)	0.685 (0.093)	0.06	0.32	0.20	0.02	0.34
CTree	5	0	1.000 (0.000)	0.993 (0.003)	0.001	0.969	0.675 (0.080)	0.627 (0.108)	0.14	0.30	0.30	0.22	0.14
CTree	0	5	0.945 (0.030)	0.890 (0.041)	0.287	0.364	0.703 (0.048)	0.645 (0.062)	0.66	0.30	0.34	0.24	0.40
CTree	5	5	0.971 (0.016)	0.943 (0.023)	0.009	0.714	0.642 (0.145)	0.649 (0.140)	0.00	0.30	0.28	0.00	0.10
CTree	10	10	0.970 (0.012)	0.944 (0.013)	0.009	0.708	0.745 (0.134)	0.733 (0.126)	0.00	0.24	0.34	0.00	0.06
CRF	0	0	1.000 (0.000)	1.000 (0.000)	0.003	0.931	0.891 (0.016)	0.615 (0.067)	0.16	0.36	0.20	0.29	0.13
CRF	1	0	1.000 (0.000)	1.000 (0.000)	0.002	0.965	0.883 (0.018)	0.632 (0.067)	0.16	0.33	0.23	0.28	0.15
CRF	0	1	1.000 (0.000)	0.993 (0.002)	0.017	0.567	0.808 (0.029)	0.706 (0.027)	0.07	0.26	0.15	0.03	0.39
CRF	1	1	1.000 (0.000)	0.993 (0.005)	0.007	0.653	0.852 (0.026)	0.713 (0.025)	0.07	0.36	0.27	0.04	0.25
CRF	5	0	1.000 (0.000)	1.000 (0.000)	0.001	0.961	0.890 (0.019)	0.635 (0.045)	0.15	0.31	0.27	0.27	0.12
CRF	0	5	1.000 (0.000)	0.997 (0.002)	0.255	0.417	0.902 (0.026)	0.685 (0.042)	0.22	0.29	0.25	0.24	0.22
CRF	5	5	0.998 (0.001)	0.973 (0.005)	0.007	0.682	0.899 (0.028)	**0.758 (0.037)**	0.03	0.28	0.33	0.00	0.12
CRF	10	10	0.996 (0.002)	0.963 (0.006)	0.006	0.695	**0.929 (0.037)**	**0.780 (0.038)**	0.03	0.24	0.32	0.00	0.10
rpart	-	-	0.999 (0.001)	0.999 (0.002)	0.007	0.941	0.520 (0.035)	0.512 (0.043)	0.00	0.24	0.04	0.20	0.02

Table 24: Average results over $N = 10$ replicates, for different invariance-based methods, including our Constrained Random Forest (CRF) with different values of λ_1 and λ_2, as well as standard random forest (RF), applied to data simulated from the scaled-up version of model (e). The table shows the AUC and the prediction accuracy for the training and test data respectively, with standard deviations in parentheses. For the training data, $\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$ and $\hat{I}(Z; h_{\mathcal{T}}(X))$ are shown. The average importance of each feature vector, averaged over each vector entry, is shown as well. The highest value of the AUC, the prediction accuracy as well as all estimates within two standard errors of these values are marked in bold.

Model (e)													
			Training				Test		Importance				
Method	λ_1	λ_2	AUC	Accuracy	$\hat{I}(Y; Z \mid h_{\mathcal{T}}(X))$	$\hat{I}(Z; h_{\mathcal{T}}(X))$	AUC	Accuracy	$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	$\boldsymbol{X}_3$	$\boldsymbol{X}_4$	$\boldsymbol{X}_5$
CRF	1	1	1.000 (0.000)	0.993 (0.005)	0.007	0.653	0.852 (0.026)	0.713 (0.025)	0.00	0.50	0.24	0.01	0.28
CRF	5	5	0.998 (0.001)	0.973 (0.005)	0.007	0.682	0.899 (0.028)	**0.758 (0.037)**	0.00	0.56	0.13	0.00	0.04
CRF	10	10	0.996 (0.002)	0.963 (0.006)	0.006	0.695	**0.929 (0.037)**	**0.780 (0.038)**	0.00	0.50	0.07	0.00	0.02
RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.818 (0.030)	0.547 (0.065)	0.00	0.77	0.24	0.34	0.41
ICP + rpart	-	-	1.000 (0.001)	0.999 (0.001)	0.003	0.971	0.514 (0.044)	0.506 (0.046)	0.00	0.77	0.02	0.44	0.00
ICP + RF	-	-	1.000 (0.000)	1.000 (0.000)	-	-	0.522 (0.067)	0.492 (0.025)	0.00	0.63	0.02	0.37	0.00
AnchorBoosting	1	-	1.000 (0.000)	1.000 (0.000)	-	-	0.507 (0.182)	0.512 (0.043)	0.00	0.19	0.04	0.13	0.02
AnchorBoosting	5	-	1.000 (0.000)	1.000 (0.000)	-	-	0.647 (0.156)	0.513 (0.071)	0.00	0.22	0.00	0.05	0.00
AnchorBoosting	10	-	1.000 (0.000)	1.000 (0.000)	-	-	0.717 (0.153)	0.503 (0.036)	0.00	0.17	0.00	0.07	0.00
IRF	1	-	0.958 (0.009)	0.837 (0.013)	-	-	**0.920 (0.038)**	0.639 (0.037)	0.03	0.00	0.65	0.03	0.74
IRF	5	-	0.962 (0.007)	0.836 (0.013)	-	-	**0.929 (0.025)**	0.622 (0.040)	0.05	0.00	0.74	0.14	0.71
IRF	10	-	0.962 (0.007)	0.837 (0.012)	-	-	**0.934 (0.024)**	0.603 (0.062)	0.07	0.00	0.77	0.21	0.68

E.3 EFFECT OF NOT SUBSAMPLING FEATURES

To further assess the effect of not performing feature subsampling in CRF, we have conducted an additional simulation study in the lower-dimensional setting of Section 5.1, using the five SCMs (a)–(e) with $p = 5$ features and 300 training and 300 test observations in each replicate. We compare the default CRF, which evaluates each split on the full set of variables, to a variant using standard RF-style feature subsampling (*mtry*) where we sample $\lfloor \sqrt{p} \rfloor$ features to consider at each split. We evaluate CRF with and without feature subsampling with $(\lambda_1, \lambda_2) \in \{(5, 5), (10, 10)\}$. In addition to test accuracy, we report three pairwise tree-diversity measures: the disagreement rate, the double-fault rate, and the Jaccard similarity of the feature sets used by pairs of trees, together with mean feature importance.

Let B denote the number of trees in the forest, let $\hat{y}_i^{(b)}$ be the prediction of tree b for test observation i, and let $S^{(b)}$ be the set of features used by tree b. We consider three pairwise diversity measures, averaged over all $\binom{B}{2}$ pairs of trees. The disagreement rate is

$$D_{\text{dis}} = \frac{1}{\binom{B}{2}} \sum_{b<b'} \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \mathbb{1}\{\hat{y}_i^{(b)} \neq \hat{y}_i^{(b')}\},$$

the double-fault rate is

$$D_{\text{df}} = \frac{1}{\binom{B}{2}} \sum_{b<b'} \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \mathbb{1}\{\hat{y}_i^{(b)} \neq y_i,\ \hat{y}_i^{(b')} \neq y_i\},$$

and the Jaccard similarity is

$$J = \frac{1}{\binom{B}{2}} \sum_{b<b'} \frac{|S^{(b)} \cap S^{(b')}|}{|S^{(b)} \cup S^{(b')}|}.$$

These measure predictive disagreement, shared errors, and overlap in the features used by pairs of trees, respectively.

The results are given in Table 25, and they show a clear tradeoff. As expected, feature subsampling leads to less structural overlap between trees, reflected in lower Jaccard similarity and higher disagreement. However, it also yields consistently lower test accuracy across all five models, with the largest losses in the more challenging settings (a), (d), and (e), and tends to produce larger double-fault rates. Moreover, the version without feature subsampling places importance more strongly

Table 25: Average results over $N = 20$ replicates, showing test data accuracy, diversity measures, and mean feature importance for the comparison between CRF without feature subsampling (*full*) and CRF with feature subsampling (*mtry*) across simulation models (a)–(e) with $n_{\text{train}} = n_{\text{test}} = 300$. The table reports prediction accuracy, disagreement rate, double-fault rate, Jaccard similarity, and mean feature importance. The best accuracy within each model as well as all estimates within two standard errors of these values are marked in bold.

			Test				Importance				
Method	λ_1	λ_2	Accuracy	Disagree	DblFault	Jaccard	X_1	X_2	X_3	X_4	X_5
Model (a)											
CRF (full)	5	5	0.889 (0.042)	0.220 (0.081)	0.090 (0.029)	0.706 (0.022)	0.30	0.99	0.89	0.01	0.63
CRF (full)	10	10	**0.906 (0.031)**	0.166 (0.077)	0.075 (0.025)	0.717 (0.021)	0.20	1.00	0.88	0.00	0.59
CRF (mtry)	5	5	0.828 (0.086)	0.379 (0.041)	0.143 (0.017)	0.517 (0.045)	0.56	0.69	0.76	0.16	0.76
CRF (mtry)	10	10	0.853 (0.091)	0.368 (0.041)	0.135 (0.019)	0.493 (0.039)	0.51	0.64	0.74	0.10	0.73
Model (b)											
CRF (full)	5	5	**0.962 (0.013)**	0.111 (0.041)	0.031 (0.006)	0.656 (0.064)	0.47	0.80	0.99	0.22	0.06
CRF (full)	10	10	**0.962 (0.011)**	0.104 (0.031)	0.032 (0.006)	0.645 (0.073)	0.49	0.76	0.99	0.16	0.06
CRF (mtry)	5	5	0.947 (0.016)	0.230 (0.039)	0.053 (0.012)	0.523 (0.037)	0.74	0.67	0.85	0.30	0.48
CRF (mtry)	10	10	0.944 (0.025)	0.207 (0.040)	0.051 (0.014)	0.517 (0.036)	0.73	0.66	0.85	0.24	0.45
Model (c)											
CRF (full)	5	5	**0.966 (0.015)**	0.100 (0.028)	0.029 (0.006)	0.674 (0.063)	0.53	0.80	0.99	0.14	0.03
CRF (full)	10	10	**0.963 (0.013)**	0.099 (0.020)	0.031 (0.007)	0.671 (0.061)	0.54	0.77	0.98	0.09	0.03
CRF (mtry)	5	5	0.947 (0.018)	0.230 (0.031)	0.051 (0.010)	0.518 (0.030)	0.75	0.67	0.84	0.26	0.45
CRF (mtry)	10	10	0.944 (0.018)	0.212 (0.028)	0.050 (0.011)	0.513 (0.033)	0.73	0.66	0.84	0.20	0.42
Model (d)											
CRF (full)	5	5	0.886 (0.041)	0.221 (0.081)	0.092 (0.032)	0.706 (0.024)	0.29	0.99	0.89	0.01	0.63
CRF (full)	10	10	**0.903 (0.034)**	0.166 (0.076)	0.075 (0.026)	0.718 (0.021)	0.19	1.00	0.88	0.00	0.59
CRF (mtry)	5	5	0.824 (0.085)	0.379 (0.042)	0.144 (0.018)	0.518 (0.045)	0.56	0.69	0.76	0.16	0.76
CRF (mtry)	10	10	0.849 (0.091)	0.368 (0.041)	0.137 (0.020)	0.493 (0.040)	0.51	0.63	0.74	0.10	0.73
Model (e)											
CRF (full)	5	5	**0.607 (0.046)**	0.263 (0.124)	0.281 (0.059)	0.706 (0.022)	0.30	0.99	0.89	0.01	0.63
CRF (full)	10	10	**0.626 (0.045)**	0.198 (0.119)	0.287 (0.065)	0.717 (0.021)	0.20	1.00	0.88	0.00	0.59
CRF (mtry)	5	5	0.582 (0.046)	0.388 (0.056)	0.266 (0.030)	0.517 (0.045)	0.56	0.69	0.76	0.16	0.76
CRF (mtry)	10	10	0.595 (0.050)	0.361 (0.053)	0.270 (0.030)	0.493 (0.039)	0.51	0.64	0.74	0.10	0.73

on the intended predictive features X_2 and X_3 and less on the spurious or environment-dependent variables such as X_4 or X_5. Overall, these results support our design choice to omit feature subsampling in CRF: although this reduces ensemble diversity relative to standard RF-style subsampling, it allows each split to evaluate the constrained objective on the full set of variables and leads to better OOD performance in our setting. Feature subsampling is, however, available as an optional feature in the CRF implementation.