Comparative Analysis of Double Machine Learning Methods and Data Generating Processes for Average Treatment Effect Estimation in Diverse Causal Scenarios

Fernando Rocha Urbano

Autumn 2024

1 Abstract

This paper aims to establish and showcase theoretical background and practical implementation through simulations of data generating processes (DGPs) and the application of Double Machine Learning (DML) methods across various causal inference scenarios, including Backdoor Adjustment, Frontdoor Adjustment, and Instrumental Variables, especially in the context and goal of estimating the Average Treatment Effect (ATE).

We start by reviewing the literature and theoretical context of Double Machine Learning and dive into double robust estimation in the presence of instrumental variables, unobserved confounders, and mediators, showcasing differences in orthogonal score functions and implications of misspecification. We follow by reviewing the DGP under each of these causal scenarios, including the assumptions of each, the role of the nuisance functions in the estimation of the Average Treatment Effect (ATE), and the range of parameters that can be defined in simulations.

Finally, we develop a robust simulation framework capable of generating data under diverse causal settings, accommodating linear and nonlinear relationships, varying dimensions and sparsity of causal and non-causal covariates, and differing noise structures. The system serves as the building block of a long-term project that can evaluate the performance of DML methods in recovering the ATE across different causal scenarios.

We present a vast range of results for the Backdoor Adjustment scenario, exploring how various DML estimators perform in recovering the ATE. Here we provide actionable recommendations of which model between DNN, Random Forest, LASSO, Elastic Net, and OLS performs better under different conditions, highlighting differences in convergence rate, noise, nuisance function specification/misspecification, sample size, sparsity of covariates, and types of causal relationship between the target, treatment, and covariates (linear and non-linear).

Contents

2 Double Machine Learning

2.1 Problem Setup

Consider a random sample $(Y_i, T_i, X_i)_{i=1}^n$, where $Y_i \in \mathbb{R}$ is the outcome variable, $T_i \in \{0, 1\}$ is a binary treatment indicator, and $X_i \in \mathbb{R}^d$ is a vector of covariates.

Our goal is to estimate the Average Treatment Effect (ATE), defined as:

$$\tau = \mathbb{E}[Y_i(1) - Y_i(0)], \tag{2.1}$$

where $Y_i(t)$ denotes the potential outcome for unit i under treatment $T_i = t$

The potential outcomes framework, introduced by Rubin (1974) and continuously developed Imbens & Rubin (2015), conceptualizes the causal effect of a binary treatment $T_i \in \{0, 1\}$ on an outcome Y_i through the unobservable pair $(Y_i(0), Y_i(1))$.

In an ideal randomized experiment, the treatment assignment T_i is independent of these potential outcomes, ensuring unbiased estimates of the Average Treatment Effect (ATE). However, in observational data, the process that assigns treatment is rarely random, and T_i often depends on observed and possibly high-dimensional covariates $X_i \in \mathbb{R}^d$.

This complicates the identification of the ATE, as one only observes $Y_i(T_i)$ for each unit i while the counterfactual $Y_i(1-T_i)$ remains unobserved. Thus, recovering the joint distribution of $(Y_i(0), Y_i(1))$ from the observed data $\{(Y_i, T_i, X_i)\}_{i=1}^n$ is fundamentally a missing data problem.

In observational data, the Conditional Independence Assumption (CIA) is a crucial aspect of the potential outcomes framework, meaning that $(Y_i(0), Y_i(1)) \perp T_i \mid X_i$ must hold for us to correctly estimate the treatment effect. Under this assumption and overlap conditions ensuring $0 < \mathbb{P}(T_i = 1 \mid X_i) < 1$, one can express:

$$\tau = \mathbb{E}[Y_i(1) - Y_i(0)] = \mathbb{E}[\mu_1(X_i) - \mu_0(X_i)], \tag{2.2}$$

where $\mu_t(X_i) = \mathbb{E}[Y_i \mid T_i = t, X_i]$. Nonetheless, consistently estimating $\mu_0(X)$, $\mu_1(X)$, and thus τ is challenging in practice.

First, any violations of CIA, including omitted confounders, wrong model specification, or measurement error in X_i, will bias the estimates. Second, even if CIA holds, correct specification of the functional forms for $\mu_0(X)$ and $\mu_1(X)$ is difficult.

In modern settings, X_i is often high-dimensional, making parametric assumptions too restrictive and prone to model misspecification. Thus, an unbiased estimation of the ATE is often challenging Chernozhukov et al. (2018a) and requires flexible, data-driven methods that can accommodate complex relationships between Y_i, T_i, and X_i.

2.2 From Linear Estimator to Double Robust Methods

Early approaches to estimating the ATE under the CIA often began with simple linear models for the outcome regression or propensity score.

$$\mu_t(X_i) = X_i^\top \beta_t \quad \text{and} \quad \pi(X_i) = X_i^\top \gamma, \tag{2.3}$$

Such linear estimators assume that $\mu_t(X_i)$ and $\pi(X_i) = \mathbb{P}(T_i = 1 \mid X_i)$ can be captured by low-dimensional linear functions of X_i. However, any deviation from linearity or additivity, or the presence of omitted interactions, can result in substantial bias and inconsistent estimates of τ Imbens & Rubin (2015).

To mitigate these issues, the literature has developed estimators that maintain consistency under weaker conditions. Chief among these are doubly robust estimators, which combine an outcome model and a propensity score model in such a way that if either one is correctly specified, the resulting ATE estimate remains consistent Robins et al. (1994); Bang & Robins (2005). In other words, double robustness relaxes the strong reliance on fully correct functional specification of the data-generating process. This property becomes particularly appealing in high-dimensional settings, where even flexible parametric models may fail to capture all relevant nonlinearities and interactions Chernozhukov et al. (2018a).

By exploiting both an outcome regression model $\mu_t(X_i)$ and a propensity score model $\pi(X_i)$, doubly robust estimators attain unbiasedness under weaker conditions than either model alone would require. Specifically, if at least one of these models is correctly specified, the doubly robust estimator remains consistent for τ Robins et al. (1994); Bang & Robins (2005). A canonical example is the Augmented Inverse Probability Weighting (AIPW) estimator Glynn & Quinn (2010):

$$\hat{\tau}_{\text{DR}} = \frac{1}{n} \sum_{i=1}^{n} \left[\frac{T_i(Y_i - \hat{\mu}_1(X_i))}{\hat{\pi}(X_i)} + \hat{\mu}_1(X_i) - \frac{(1 - T_i)(Y_i - \hat{\mu}_0(X_i))}{1 - \hat{\pi}(X_i)} - \hat{\mu}_0(X_i) \right]. \tag{2.4}$$

In this construction, consistency does not hinge on both models being perfectly specified: correct specification of either $\hat{\mu}_t(X_i)$ or $\hat{\pi}(X_i)$ suffices to ensure unbiased estimation of τ.

While traditional doubly robust estimators often use parametric models for $\mu_t(X_i)$ and $\pi(X_i)$, DML explicitly incorporates machine learning techniques often able to capture complex relationships between Y_i, T_i, and X_i without imposing strong parametric assumptions.

2.3 DML Estimator

The DML framework aims to estimate τ while controlling for high-dimensional or complex relationships between Y_i, T_i, and X_i. The key idea is to use machine learning methods to estimate the nuisance parameters and then construct an estimator for τ that is robust to estimation errors in these nuisance parameters.

2.3.1 Nuisance Parameter Estimation

We define the following nuisance functions:

$$m(X_i) = \mathbb{E}[Y_i|X_i], \tag{2.5}$$
$$g(X_i) = \mathbb{E}[T_i|X_i], \tag{2.6}$$
$$\pi(X_i) = \mathbb{P}(T_i = 1|X_i) \quad \text{(propensity score)} \tag{2.7}$$

Such functions $\{m(X_i), g(X_i), \pi(X_i)\}$ are called nuisance parameters because, while they characterize key components of the data-generating process, they are not directly of inferential interest for τ. Instead, they serve as building blocks that allow one to express the ATE through moment equations whose solutions return consistent and asymptotically normal estimators when these functions are known or well-estimated Chernozhukov et al. (2018a); Newey (1990); Robins et al. (1994).

In practice, $\pi(X_i)$, the propensity score, controls for confounding by accounting for how units self-select into treatment based on their covariates. The functions $m(X_i) = \mathbb{E}[Y_i \mid X_i]$ and $g(X_i) = \mathbb{E}[T_i \mid X_i]$ capture the marginal relationships of the outcome and treatment with the covariates.

This property is achieved through Neyman-orthogonalization, ensuring that small perturbations in $\hat{m}(X_i), \hat{g}(X_i), \hat{\pi}(X_i)$ do not accumulate into large biases in $\hat{\tau}$ Chernozhukov et al. (2018a); Belloni et al. (2014).

These functions can be estimated using flexible machine learning methods such as LASSO, Random Forests, or Neural Networks. Under mild regularity conditions and appropriate rates of convergence for the nuisance estimators, the DML estimator is root-n consistent and asymptotically normal (we make the regularity conditions explicity in Appendix 8.1).

2.3.2 Orthogonal Score Function

To achieve robustness, we construct an orthogonal score function $\psi(Y_i, T_i, X_i; \eta)$, where $\eta = (m, g)$ represents the nuisance parameters. The orthogonal score satisfies the Neyman-orthogonality condition previously mentioned, which ensures that small estimation errors in η have a negligible first-order impact on the estimation of τ.

A common choice for the orthogonal score is for $T_i \in \{0, 1\}$ and $Y_i \in \mathbb{R}$ is:

$$\psi(Y_i, T_i, X_i; \eta) = \left(\frac{T_i - g(X_i)}{\pi(X_i)(1 - \pi(X_i))} \right) (Y_i - m(X_i)) + (m_1(X_i) - m_0(X_i)) - \tau, \tag{2.8}$$

We derive the orthogonal score function in Section 5.1 (Chernozhukov et al. (2018b)).

where $g(X_i) = \pi(X_i)$ and $m_t(X_i) = \mathbb{E}[Y_i | T_i = t, X_i]$ for $t \in \{0, 1\}$.

2.3.3 Estimation Procedure

The estimation of DML models contains several steps:

i. Estimate Nuisance Functions: Use one of the outlined ML models to obtain estimators $\hat{m}(X_i)$ and $\hat{g}(X_i)$ for the nuisance functions.

ii. Compute the Score Function: Evaluate the orthogonal score $\psi(Y_i, T_i, X_i; \hat{\eta})$ using the estimated nuisance parameters.

iii. Estimate τ: Solve the empirical moment condition which yields $\hat{\tau}$:

$$\frac{1}{n} \sum_{i=1}^{n} \psi(Y_i, T_i, X_i; \hat{\eta}) = 0. \tag{2.9}$$

The orthogonal score function mitigates the impact of errors in $\hat{m}(X)$ and $\hat{g}(X)$ on the estimation of τ while also accomodating the use of modern ML techniques, allowing complex relationships between Y, X and T.

To prevent overfitting while estimating the nuisance function and ensure that the estimation errors in the parameters do not bias the estimators of τ, it is a common practice to employ cross-fitting.

2.3.4 Cross-Fitting

The steps of cross-fitting are:

1. Split the Sample: Divide the data into K folds $\{\mathcal{I}_k\}k = 1^K$.
2. For Each Fold:
   (a) Train Nuisance Estimators: Use data from all other folds:
   $$\mathcal{I}_{-k} = \bigcup j \neq k\mathcal{I}_j$$
   to estimate $\hat{m}^{(-k)}(X_i)$ and $\hat{g}^{(-k)}(X_i)$.
   (b) Compute Score Function: For observations in fold $\mathcal{I}_k$, compute $\psi(Y_i, T_i, X_i; \hat{\eta}^{(-k)})$ using the nuisance estimates from step (a):

$$\begin{aligned}\psi(Y_i, T_i, X_i; \hat{\eta}^{(-k)}) = &\left(\frac{T_i - \hat{g}^{(-k)}(X_i)}{\hat{\pi}^{(-k)}(X_i)}\right)(Y_i - \hat{m}^{(-k)}(X_i)) \\ &+ \hat{m}_1^{(-k)}(X_i) - \hat{m}_0^{(-k)}(X_i) - \tau^{(-k)}.\end{aligned} \tag{2.10}$$

3. Aggregate: Combine the estimates from all folds:

$$\hat{\tau} = \frac{1}{n}\sum_{k=1}^{K}\sum_{i\in\mathcal{I}_k}\hat{\tau}_i^{(-k)} = \frac{1}{K}\sum_{k=1}^{K}\hat{\tau}^{(-k)} \tag{2.11}$$

3 Causal Inference Scenarios

The three most relevant scenarios of causal inference are the ones that require Instrumental Variables, Backdoor Adjustment, or Front Door Adjustment.

To facilitate the understandig of the causal scenarios, we present the following notation, separating covariates in three different groups:

- $C_i \in \mathbb{R}^{d_c}$: Confounders, variables that affect both the treatment T_i and the outcome Y_i.
- $X_a \in \mathbb{R}^{d_a}$: Non-causal covariates, variables that do not affect the treatment T_i or the outcome Y_i, but are correlated with the confounders C_i and lead to misspecification when included in the model.

- $U_i \in \mathbb{R}^{d_u}$: Unobserved causal confounders, variables that affect both the treatment T_i and the outcome Y_i but are not observed. The non inclusion of U_i in the model leads to misspecification and biased estimates of the causal effect unless a correction is present (IV scenario).

In the specified notation, CIA (2.2) can be expressed as:

$$Y_i(0), Y_i(1) \perp T_i \mid C_i, U_i \tag{3.1}$$

A common way to represent the causal relation in the three scenarios previously mentioned is through the use of DAGs Pearl (1995).

3.1 DAG (Directed Acyclic Graph)

Directed Acyclic Graph (DAG) serves as a representation of causal assumptions and a tool for deriving statistical properties of the variables involved.

It is composed of nodes (vertices) representing random variables or features and directed edges (arrows), indicating a direct influence or causal effect (in 1, a causal effect of T on Y).

Figure 1: DAG Example

The acyclic characterist is due to absence of directed cycles; that is, there is no path where you can start at a node X and, by following directed edges, return to X.

3.2 Backdoor Adjustment

A backdoor path from treatment T to the outcome Y represents alternative routes through which association can flow from T to Y that are not due to the causal effect of T on Y. In the representation, the confounder C is a common cause of both T and Y, thus, a backdoor path.

In such case, the association between T and Y may be partially or entirely due to their mutual dependence on C rather than a direct causal effect, leading to biased causal estimates of the treatment if C is ignored.

The causal effect of T on Y can be expressed using the backdoor adjustment formula:

$$\mathbb{P}[Y(t)] = \sum_{C} \mathbb{P}[Y \mid T, C]\mathbb{P}[C], \tag{3.2}$$

Which serves a markov factorization, calculating with respect to the DAG structure.

The scenario is the most commonly represented in causal inference, since it is present in most observational studies (where the treatment is not randomly assigned). It is often not even mentioned as a scenario, but as the standard case and already contains considerable part of its explanation in the previous sections.

For the backdoor adjustment to be valid, the following conditions must be satisfied:

1. No variable in the adjustment set C is a descendant of the treatment T.
2. The adjustment set C blocks all backdoor paths from T to Y (backdoor path is any path from T to Y that starts with an arrow into T).

DAG 2 illustrates the backdoor path involving the confounder C, treatment T, outcome Y, and features with non-causal association X_a which would not be present in a typical backdoor adjustment DAG, but play a relevant role in the simulations proposed.

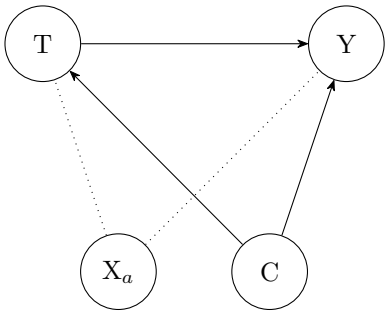


Figure 2: DAG of Backdoor Path

3.2.1 Backdoor Adjustment Scenario in DML

In Backdoor Adjustment with DML, the ideal $m(X_i) = \mathbb{E}[Y_i|X_i]$ and $g(X_i) = \mathbb{E}[T_i|X_i]$ from Equations (2.5) and (2.6) are:

$$m_{\text{BA}}(C_i) = \mathbb{E}[Y_i|C_i], \tag{3.3}$$
$$g_{\text{BA}}(C_i) = \mathbb{E}[T_i|C_i], \tag{3.4}$$

In our simulations we also test the habilit of the ML models to estimate ATE under the following scenarios. In these scenarios, the superscript w_i on the nuisance functions denotes the i-th scenario which involves misspecifications of the nuisance functions.

Inclusion of C_i and $X_{a,i}$ on every nuisance function

$$m^{\text{w}_1}_{\text{BA}}(C_i, X_{a,i}) = \mathbb{E}[Y_i|C_i, X_{a,i}], \tag{3.5}$$
$$g^{\text{w}_1}_{\text{BA}}(C_i, X_{a,i}) = \mathbb{E}[T_i|C_i, X_{a,i}], \tag{3.6}$$

Equations (3.5) and (3.6) represent a scenario in which one would not be aware that X_a does not cause T and Y.

Only partial inclusion of C_i and inclusion of $X_{a,i}$ on every nuisance function

$$m_{\mathrm{BA}}^{\mathrm{w}_2}(C_i^p, X_{a,i}) = \mathbb{E}[Y_i|C_i^p, X_{a,i}], \tag{3.7}$$
$$g_{\mathrm{BA}}^{\mathrm{w}_2}(C_i^p, X_{a,i}) = \mathbb{E}[T_i|C_i^p, X_{a,i}], \tag{3.8}$$

Equations (3.7) and (3.8) also represent a scenario in which one would not be aware that X_a does not cause T and Y and also does not include all causal cofounders. C_i^p is subset of C_i included, where $C_i \in \mathbb{R}^{d_c}$ and $C_i^p \in \mathbb{R}^{d_{cp}}$ and $d_c > d_{cp}$.

3.3 Frontdoor Adjustment

Frontdoor adjustment is a method used to estimate the causal effect of treatment T on outcome Y when there is unmeasured confounding that cannot be addressed using backdoor adjustment. It leverages a mediator M that lies on the causal path from T to Y.

The causal effect of T on Y can be expressed using the frontdoor adjustment formula:

$$\mathbb{P}[Y(t)] = \sum_{M} \mathbb{P}[M \mid T = t] \sum_{t'} \mathbb{P}[Y \mid M, T = t']\mathbb{P}[T = t']. \tag{3.9}$$

t' acts as a dummy variable of integration (or summation) which, in our case, assumes values in $\{0, 1\}$. The $\mathbb{P}[Y(t)]$ is computed by marginalizing over both the mediator M and any variation in the treatment T Pearl (2012)

For instance, the average treatment effect in case $M, T \in \{0, 1\}$:

$$\tau = [\mathbb{P}(M = 1 \mid T = 1) - \mathbb{P}(M = 1 \mid T = 0)] \times [\mathbb{E}[Y \mid M = 1] - \mathbb{E}[Y \mid M = 0]] \tag{3.10}$$

For the frontdoor adjustment to be valid, the following conditions must be satisfied:

1. All causal paths from T to Y pass through M (i.e., there is no direct effect of T on Y bypassing M).
2. There are no unmeasured confounders between T and M.
3. All backdoor paths from M to Y are blocked by T (i.e., there are no unmeasured confounders between M and Y that are not affected by T).

DAG 3 illustrates the frontdoor adjustment involving the treatment T, mediator M, outcome Y, observed confounders C and features with no causal association X_a. C and X_a would not be present in a typical frontdoor adjustment DAG, but are included due to their relevance in the proposed simulations.

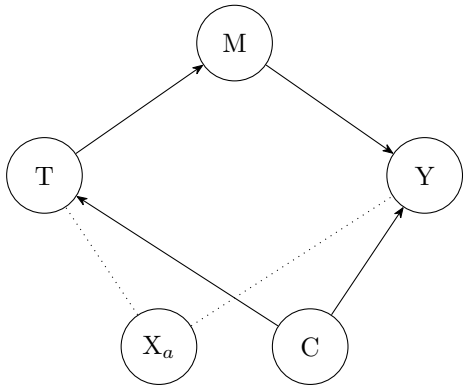


Figure 3: DAG of Frontdoor Path

3.3.1 Frontdoor Adjustment Scenario in DML

In our simulations, we use a binary mediator $M_i \in \{0, 1\}$, similar to the binary treatment T_i.

In the Frontdoor Adjustment scenario with DML, we need to account for the mediator M_i when estimating the ATE. The identification of the causal effect involves modeling the relationships between T_i, M_i, and Y_i.

The ideal nuisance functions for DML in this scenario are:

$$m_{\text{FA}}(M_i, C_i) = \mathbb{E}[Y_i \mid M_i, C_i], \tag{3.11}$$
$$h_{\text{FA}}(T_i) = \mathbb{E}(M_i \mid T_i), \tag{3.12}$$
$$g_{\text{FA}}(C_i) = \mathbb{E}(T_i \mid C_i), \tag{3.13}$$

Here $h_{\text{FA}}(M_i, T_i, C_i)$ is the mediator model, representing the probability of the mediator given treatment.

To adapt the orthogonal score function in the presence of the mediator, we modify Equation (2.8) to incorporate the mediator's effect ((Chernozhukov et al. (2018b))). The adapted orthogonal score function is:

$$\psi(Y_i, T_i, M_i, C_i; \eta) = \left(\left(\frac{M_i - h_{\text{FA}}(T_i)}{h_{\text{FA}}(T_i)(1 - h_{\text{FA}}(T_i))}\right)(Y_i - m_{\text{FA}}(M_i, C_i))\right) \cdot (h_{\text{FA}}(T_i = 1) - h_{\text{FA}}(T_i = 0)) - \tau \tag{3.14}$$

In this score function:

- The first term adjusts for the probability of the mediator and its interaction with the outcome.
- The second term adjusts for the probability of the mediator given treatment.

We derive the orthogonal score function for Frontdoor Adjustment in Section **??**.

In our simulations we also test the hability of the ML models to estimate ATE under the following scenario.

Inclusion of C_i and $X_{a,i}$ on every nuisance function

$$m_{\text{FA}}^{\text{w}_1}(M_i, C_i, X_{a,i}) = \mathbb{E}[Y_i \mid M_i, C_i, X_{a,i}], \tag{3.15}$$
$$h_{\text{FA}}^{\text{w}_1}(T_i, C_i, X_{a,i}) = \mathbb{E}(M_i \mid C_i, X_{a,i}), \tag{3.16}$$
$$g_{\text{FA}}^{\text{w}_1}(C_i, X_{a,i}) = \mathbb{E}(T_i \mid C_i, X_{a,i}), \tag{3.17}$$

Equations (3.15), (3.16), and (3.17) represent a scenario where one is unaware that $X_{a,i}$ has no causal relation to T_i, M_i, or Y_i. Equation (3.16) represents scenarion where one is unaware that C has no causal relation to M_i.

Ignoring the mediator M_i

$$m_{\text{FA}}^{\text{w}_2}(C_i, X_{a,i}) = \mathbb{E}[Y_i \mid C_i, X_{a,i}], \tag{3.18}$$
$$g_{\text{FA}}^{\text{w}_2}(C_i, X_{a,i}) = \mathbb{E}[T_i \mid C_i, X_{a,i}], \tag{3.19}$$

Equations (3.18) and (3.19) represent a scenario where the mediator M_i is unavailable or ignored and one is unaware that $X_{a,i}$ has no causal relationship to T or Y. In this case, the orthogonal score function is the same as in Equation (2.8).

Considering the mediator M_i as a normal covariate

$$m_{\text{FA}}^{\text{w}_3}(C_i, M_i, X_{a,i}) = \mathbb{E}[Y_i \mid C_i, M_i, X_{a,i}], \tag{3.20}$$
$$g_{\text{FA}}^{\text{w}_3}(C_i, M_i, X_{a,i}) = \mathbb{E}[T_i \mid C_i, M_i, X_{a,i}], \tag{3.21}$$

Equations (3.20) to (3.21) also represent scenarios where one is unaware that $X_{a,i}$ has no causal relation to T_i or Y_i, and the scenario where M_i is available but not recognized as a mediator. In this case, the orthogonal score function is the same as in Equation (2.8).

3.4 Instrumental Variable

Instrumental variable (IV) estimation is a method used to estimate the causal effect of a treatment T_i on an outcome Y_i when there is unmeasured confounding U_i that cannot be addressed using backdoor or frontdoor adjustments. This method leverages an instrument Z_i, which influences the treatment T_i but has no direct effect on the outcome Y_i except through T_i, and is independent of any unmeasured confounders U_i affecting both T_i and Y_i.

For the instrumental variable method to be valid, the following conditions must be satisfied:

1. Relevance: The instrument Z is associated with the treatment T (i.e., $\text{Cov}(Z, T) \neq 0$ or $Z \not\perp\!\!\!\perp T$).

2. Exclusion Restriction: The instrument Z affects the outcome Y only through its effect on the treatment T (i.e., there is no direct effect of Z on Y and no other pathways from Z to Y except through T).

3. Independence (Ignorability): The instrument Z is independent of any unmeasured confounders U that affect both T and Y ($i.e., Z \perp\!\!\!\perp U$).

DAG 4 illustrates the instrumental variable setup involving the unobserved confounder U, instrument Z, treatment T, outcome Y, observed confounder C, and features with non-causal associations X_a. Again, the last two would not be present in a typical instrumental variable DAG, but are included due to their relevance in the proposed simulations.

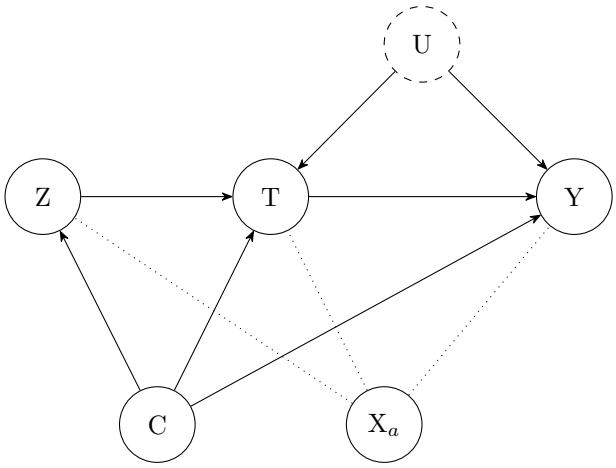


Figure 4: DAG of Instrumental Variable

3.4.1 Instrumental Variable Scenario in DML

To adapt the DML framework to the IV setting, we need to define appropriate nuisance functions and construct an orthogonal score function suitable for the IV context.

The ideal nuisance functions in this scenario are:

$$m_{\text{IV}}(C_i) = \mathbb{E}[Y_i \mid C_i], \tag{3.22}$$
$$q_{\text{IV}}(C_i) = \mathbb{E}[Z_i \mid C_i], \tag{3.23}$$
$$g_{\text{IV}}(C_i) = \mathbb{E}[T_i \mid C_i], \tag{3.24}$$

Where $m_{\text{IV}}(C_i)$ is the outcome model, capturing the expected outcome given covariates, $q_{\text{IV}}(C_i)$ is the instrument propensity score, representing the expected treatment given the instrument and covariates, and $g_{\text{IV}}(C_i)$ is the treatment model, representing the expected treatment given covariates.

The orthogonal score function for the IV scenario is different from the standard DML orthogonal score function (Belloni et al. (2014); Chernozhukov et al. (2018b)). An appropriate orthogonal score function in the linear IV context is:

$$\psi(Y_i, T_i, Z_i, C_i; \tau, \eta) = (Y_i - m_{\text{IV}}(C_i) - \tau[T_i - g_{\text{IV}}(C_i)])\,(Z_i - q_{\text{IV}}(C_i))\,, \tag{3.25}$$

We derive the orthogonal score function for Frontdoor Adjustment in Section 5.3.

In our simulations we also test the habilit of the ML models to estimate ATE under the following scenarios. In these scenarios, the superscript w_i on the nuisance functions denotes the i-th scenario which involves misspecifications of the nuisance functions.

Inclusion of C_i and $X_{a,i}$ on every nuisance function

$$m_{\text{IV}}^{\text{w}_1}(C_i) = \mathbb{E}[Y_i \mid C_i, X_{a,i}], \tag{3.26}$$
$$q_{\text{IV}}^{\text{w}_1}(C_i) = \mathbb{E}[Z_i \mid C_i, X_{a,i}], \tag{3.27}$$
$$g_{\text{IV}}^{\text{w}_1}(C_i) = \mathbb{E}[T_i \mid C_i, X_{a,i}], \tag{3.28}$$

Equations (3.26), (3.27), and (3.28) represent a scenario in which one would not be aware that X_a does not cause T, Z and Y.

Only partial inclusion of C_i and inclusion of $X_{a,i}$ on every nuisance function

$$m_{\text{IV}}^{\text{w}_2}(C_i^p, X_{a,i}) = \mathbb{E}[Y_i \mid C_i, X_{a,i}], \tag{3.29}$$
$$q_{\text{IV}}^{\text{w}_2}(C_i^p, X_{a,i}) = \mathbb{E}[Z_i \mid, C_i, X_{a,i}], \tag{3.30}$$
$$g_{\text{IV}}^{\text{w}_2}(C_i^p, X_{a,i}) = \mathbb{E}[T_i \mid C_i, X_{a,i}], \tag{3.31}$$

Equations (3.29), (3.30), and (3.8) also represent a scenario in which one would not be aware that X_a does not cause T and Y and also does not include all causal cofounders. C_i^p is subset of C_i included, where $C_i \in \mathbb{R}^{d_c}$ and $C_i^p \in \mathbb{R}^{d_{cp}}$ and $d_c > d_{cp}$.

Z_i is treated as a normal cofounder

$$m_{\text{IV}}^{\text{w}_3}(C_i, Z_i) = \mathbb{E}[Y_i | C_i, Z_i], \tag{3.32}$$
$$g_{\text{IV}}^{\text{w}_3}(C_i, Z_i) = \mathbb{E}[T_i | C_i, Z_i], \tag{3.33}$$

Equations (3.34) and (3.35) represent a scenario in which one would treat Z as a normal covariate. In this case, the orthogonal score function would be the same as in the backdoor adjustment scenario of (2.8).

Z_i is treated as a normal cofounder and inclusion of $X_{a,i}$ on every nuisance function

$$m_{\text{IV}}^{\text{w}_4}(C_i, Z_i, X_{a,i}) = \mathbb{E}[Y_i | C_i, Z_i, X_{a,i}], \tag{3.34}$$
$$g_{\text{IV}}^{\text{w}_4}(C_i, Z_i, X_{a,i}) = \mathbb{E}[T_i | C_i, Z_i, X_{a,i}], \tag{3.35}$$

Equations (3.34) and (3.35) represent a scenario in which one would treat Z as a normal covariate and is not aware that X_a does not cause T, Z and Y. In this case, the orthogonal score function would also be the same as in the backdoor adjustment scenario of (2.8).

4 Simulation Methodology

4.1 Data Generating Process

We define the the backdoor path scenario as the typical scenario. The frontdoor path and instrumental variable scenarios are variations of the typical scenario.

4.1.1 Cofounders

The cofounders are generated with multivariate normal distribution using a sparse covariance matrix Σ_d and a vector of means $\boldsymbol{\mu}_d = \mathbf{0}$.

$\Sigma_d \in \mathbb{R}^{d \times d}$, where $d := d_a + d_c$. d_a is the number of non-causal confounders previously represented as X_a and d_c is the number of causal confounders previously represented as C.

$$X \sim \mathcal{N}_d(\boldsymbol{\mu}_d, \Sigma_d) \tag{4.1}$$

The sparsity of the covariance matrix of the cofounders Σ_d is determined by α_d, which represents the probability that any covariance coefficient is zero. Larger α_d's result in sparser covariance matrices. To provide intuition on it, we report the average absolute correlation $|\rho_d|$ in the matrix.

After generating the data from the multivariate normal distribution defined as $X \in \mathbb{R}^{n \times d}$ we randomly separate the features of X in X_a and C. Following the logic specified above, $X_a \in \mathbb{R}^{n \times d_a}$ and $C \in \mathbb{R}^{n \times d_c}$.

In (3.7) and (3.8) "Only partial inclusion of C_i and inclusion of $X_{a,i}$ on every nuisance function", after generating cofounders, treatment, and target, we select a subset of the causal confounders C_i^p to be used in the nuisance functions. This would represent a scenario of unobserved confounders U_i. We define which causal confounders are not included in the nuisance function randomly, through the use of p_u, which represents the percentage of causal confounders not included in the nuisance functions. Meaning that $C_i^p \in \mathbb{R}^{n \times d_{cp}}$, where $d_{cp} = d_c - \lceil p_u \cdot d_c \rceil$.

4.1.2 Treatment

The probability of treatment is generated from a logic:

$$\mathbb{P}[T_i] = \frac{1}{1 + e^{(-f(C_i) + \varepsilon_{t,i})}} \tag{4.2}$$

where $f(C_i)$ is a function of the causal confounders C_i and $\varepsilon_{t,i}$ is a random error term generated from a normal distribution with mean 0 and variance σ_t.

$f(C_i)$ is a generic non-linear function explained in Section 4.1.5.

Treatment is later generated from a Bernoulli distribution with the probability of treatment from above in (4.2).

$$T_i \sim Bernoulli(\mathbb{P}[T_i]) \tag{4.3}$$

4.1.3 Treatment Effect

The true individual treatment effect is generated from:

$$\tau_i(C_i) = f(C_i) \tag{4.4}$$

Again, $f(C_i)$ is a generic non-linear function explained in Section 4.1.5.

Meaning that there is heterogeneity in the treatment effect across the causal confounders C_i.

The true ATE (Average Treatment Effect) is the average of the treatment effect across the population:

$$\tau = \mathbb{E}[\tau(C_i)] = \frac{1}{n}\sum_{i=1}^{n} \tau_i(C_i) \tag{4.5}$$

4.1.4 Outcome

The observed outcome for each individual is:

$$Y_i = Y_{i,0} + \tau_i(C_i) \cdot T_i + \varepsilon_{y,i}, \quad \varepsilon_{y,i} \sim \mathcal{N}(0, \sigma_y) \tag{4.6}$$

where $Y_{i,0}$ is the potential outcome if the treatment T_i was not applied, $\tau_i(C_i)$ is the treatment effect, and $\varepsilon_{y,i}$ is a random error term generated from a normal distribution with mean 0 and variance σ_y.

$$Y_{i,0} = f(C_i) \tag{4.7}$$

Once more, $f(C_i)$ is a generic non-linear function explained in Section 4.1.5.

The potential outcome for treatment and control are:

$$Y_i(1) = Y_{i,0} + \tau_i(C_i), \quad Y_i(0) = Y_{i,0}, \tag{4.8}$$

which can not be directly observed.

4.1.5 $f(C_i)$ Non-Linear Transformation

The function $f(C_i)$, used to calculate $Y_{i,0}$ and $\mathbb{P}[T_i]$ is a generic function composed non-linear relationships.

More specifically $f(C_i)$ is the weighted sum of of different "q" transformations in the causal confounders C_i:

$$f(C) = \sum_{j=1}^{d_c} (\beta_j \cdot q_j(C_j)) \tag{4.9}$$

Where C_j is the the vector of the j-th causal confounder, β_j is a random coefficient drawn from a uniform distribution $U(-1, 1)$, and $q_j(C_j)$ is a random transformation of the j-th causal confounder. More specifically, the chosen transformation $q_j(C_j)$ is randomly chosen from the following set of possible transformations:

i. Linear: $q_j(C_{i,j}) = C_{i,j}$

ii. Quadratic: $q_j(C_{i,j}) = C_{i,j}^2$

iii. Cubic: $q_j(C_{i,j}) = C_{i,j}^3$

iv. Logarithmic: $q_j(C_{i,j}) = \log(|C_{i,j}| + 1)$

v. Exponential: $q_j(C_{i,j}) = \exp(\frac{1}{5}C_{i,j})$

vi. Sine: $q_j(C_{i,j}) = \sin(C_{i,j})$

vii. Cosine: $q_j(C_{i,j}) = \cos(C_{i,j})$

viii. Indicator: $q_j(C_{i,j}) = \mathbb{I}\{C_{i,j} > 0\} - \mathbb{I}\{C_{i,j} \leq 0\}$

ix. Piecewise: $q_j(C_{i,j}) = 2\mathbb{I}\{C_{i,j} < 0\} + \mathbb{I}\{0 \leq C_{i,j} < 1\} + \frac{1}{2}\mathbb{I}\{C_{i,j} > 1\}$

For each simulation, $f(C)$ is calculated with some of the specified transformation sets and β's three times: one for the calculation of $Y_{i,0}$, one for the calculation of $\mathbb{P}[T_i]$, and one for the calculation of $\tau_i(C_i)$.

For some simulations, we only allow for linear transformations in $f(C)$, aiming to compare performance of the DML models in a simpler setting.

4.1.6 Parameter Tuning in the Data Generating Process

The data generating process has some parameters that can be tuned to generate different scenarios. We present the following variations:

i. $f(C_i)$ for Treatment Probability: linear or non-linear transformations (4.2).

ii. $f(C_i)$ for Treatment Effect: linear or non-linear transformations (4.4).

iii. $f(C_i)$ for Outcome: linear or non-linear transformations (4.7).

iv. d_c: number of causal confounders.

v. d_a: number of non-causal confounders.

vi. p_u: percentage of causal confounders not included in the nuisance functions in the backdoor adjustment scenario with wrong specification (3.7, 3.8).

vii. n: number of observations.

viii. α_d: sparsity of the covariance matrix of cofounders (X), which is a direct cause of average absolute correlation between cofounders $|\rho_d|$ (4.1).

ix. σ_t: variance of the error term ε_t in the treatment probability (4.2).

x. σ_y: variance of the error term ε_y in the outcome (4.6).

4.2 Differences in Data Generating Process for Frontdoor Scenario

The frontdoor scenario introduces some modifications to the data generating process relative to the backdoor adjustment scenario. While the backdoor scenario considers a direct effect of the treatment T on the outcome Y (possibly influenced by C), the frontdoor scenario leverages a mediator M that lies on the causal pathway from T to Y. Thus, the presence of M alters the structure in which the outcome Y is generated, as well as how T and Y relate to C.

4.2.1 Cofounders

As in the backdoor scenario, the confounders $C \in \mathbb{R}^{n \times d_c}$ are still generated from a multivariate normal distribution using a sparse covariance matrix Σ_d and a vector of means $\boldsymbol{\mu}_d = \mathbf{0}$, as described in (4.1). The set of non-causal covariates $X_a \in \mathbb{R}^{n \times d_a}$ remains the same, being drawn from the same distribution as C but without causal influence on treatment, mediator, or outcome.

As in the backdoor scenario, the role of X_a is to serve as non-causal but potentially confounding features when mistakenly included in the nuisance functions. Unlike the instrumental variable scenario, we do not introduce unobserved confounders U or an instrument Z here. Instead, the key difference is the introduction of the mediator M.

4.2.2 Mediator

In the frontdoor scenario, the mediator M_i lies on the causal path from the treatment T_i to the outcome Y_i. This mediator is generated as a function of the treatment T_i (and potentially noise), but not directly from C or X_a:

$$\mathbb{P}[M_i] = \frac{1}{1 + e^{(-f(T_i) + \varepsilon_{m,i})}}, \quad \varepsilon_{m,i} \sim \mathcal{N}(0, \sigma_m) \tag{4.10}$$

where $f(T_i)$ is a function of the treatment. For binary mediators, M_i can be drawn from a Bernoulli distribution with probability (4.10):

$$M_i \sim Bernoulli(\mathbb{P}[M_i]) \tag{4.11}$$

This differs from the backdoor scenario, which does not involve a mediator and thus defines the outcome Y directly from T and C. Here, M explicitly mediates the relationship between T and Y.

4.2.3 Treatment

The treatment T_i remains a function of C_i, similar to the backdoor scenario:

$$\mathbb{P}[T_i] = \frac{1}{1 + e^{(-f(C_i) + \varepsilon_{t,i})}}, \quad \varepsilon_{t,i} \sim \mathcal{N}(0, \sigma_t) \tag{4.12}$$

and

$$T_i \sim Bernoulli(\mathbb{P}[T_i]). \tag{4.13}$$

This mirrors the backdoor scenario's approach to treatment generation, maintaining T_i as a function of C_i. The introduction of M_i does not alter how T_i is generated, only how it ultimately influences Y_i.

4.2.4 Outcome

In the backdoor scenario, the outcome Y_i depends directly on T_i and C_i. In the frontdoor scenario, the presence of M_i changes this relationship. The outcome now depends on both the mediator M_i and the confounders C_i:

$$Y_{i,0} = f(C_i), \tag{4.14}$$

$$Y_i = Y_{i,0} + \tau_i(C_i) \cdot M_i + \varepsilon_{y,i}, \quad \varepsilon_{y,i} \sim \mathcal{N}(0, \sigma_y). \tag{4.15}$$

Here, $\tau_i(C_i)$ represents how the mediator M_i translates the influence of the treatment into changes in Y_i. Notably, Y_i is no longer directly determined by T_i as in the backdoor scenario. Instead, T_i first influences M_i, and M_i subsequently affects Y_i. This is a fundamental structural difference that characterizes the frontdoor scenario.

4.2.5 Parameter Tuning in the Data Generating Process for Frontdoor Scenario

In addition to all the parameters mentioned for the backdoor scenario (e.g., d_c, d_a, n, σ_t, σ_y, and the choice of linear or non-linear transformations in $f(\cdot)$), the frontdoor scenario also introduces parameters related to the mediator generation:

i. $f(T_i)$ for Mediator Probability: linear or non-linear transformations in (4.10).

ii. σ_m: variance of the error term ε_m in the mediator generation (4.10).

By adjusting these parameters, we can create different mediation structures and assess how well the models estimate the causal effect when a mediator is present and when non-causal features X_a are included or omitted.

4.3 Differences in Data Generating Process for Instrumental Variable Scenario

The instrumental variable scenario also provides some differences in the data generating process compared to the backdoor adjustment scenario and the frontdoor adjustment scenario, thus also allowing for more parameters variation. We present the following variations in comparison to the backdoor adjustment scenario.

4.3.1 Cofounders

The cofounders are again generated from a multivariate normal distribution using a sparse covariance matrix Σ_d and a vector of means $\boldsymbol{\mu}_d = \mathbf{0}$, as described in (4.1).

In this scenario, X is divided in $X_a \in \mathbb{R}^{n \times d_a}$, $C \in \mathbb{R}^{n \times d_c}$, and $U \in \mathbb{R}^{n \times d_u}$, where $d := d_a + d_c + d_u$.

U_i is used to generate $\mathbb{P}[T_i]$, and $\tau_i(C_i, U_i)$, and is not included in the nuisance functions.

4.3.2 Instrument

We define the instrument in both discrete and continuous forms. In both cases, the instrument is generated from C_i but not from U_i.

In the discrete case:

$$\mathbb{P}[Z_i] = \frac{1}{1 + e^{(-f(C_i) + \varepsilon_{z,i})}}, \quad \varepsilon_{z,i} \sim \mathcal{N}(0, \sigma_z) \tag{4.16}$$

$$Z_i \sim Bernoulli(\mathbb{P}[Z_i]) \tag{4.17}$$

In the continuous case:

$$Z_i = f(C_i) + \varepsilon_{z,i}, \quad \varepsilon_{z,i} \sim \mathcal{N}(0, \sigma_z) \tag{4.18}$$

4.3.3 Treatment

The $\mathbb{P}[T_i]$ previously addressed in the backdoor adjustment scenario with (4.2) becomes:

$$\mathbb{P}[T_i] = \frac{1}{1 + e^{(-f(C_i, U_i, Z_i) + \varepsilon_{t,i})}}, \quad \varepsilon_{t,i} \sim \mathcal{N}(0, \sigma_t) \tag{4.19}$$

A similar procedure is done for the individual treatment effect from (4.4):

$$\tau_i(C_i, U_i) = f(C_i, U_i) \tag{4.20}$$

In a standard instrumental variables (IV) framework, the key assumption is that the instrument Z affects the outcome Y only through its influence on the treatment T, thus influencing probability of treatment, but having no effect on the potential outcomes.

4.3.4 Outcome

Differently from the treatment, the data generating process of the outcome is not a function of the instrument Z_i. Nonetheless, it is influenced by the unobserved confounders U_i, thus still having a data generating process different from the backdoor adjustment scenario (4.7).

$$Y_{i,0} = f(C_i, U_i) \tag{4.21}$$
$$Y_i = Y_{i,0} + \tau_i(C_i, U_i) \cdot T_i + \varepsilon_{y,i}, \quad \varepsilon_{y,i} \sim \mathcal{N}(0, \sigma_y) \tag{4.22}$$

4.3.5 Parameter Tuning in the Data Generating Process for Instrumental Variable Scenario

In the instrumental variable scenario, we have the possibility to tune all the previously mentioned parameters in (4.3) as well as:

i. $f(C_i)$ for Instrument: linear or non-linear transformations (4.16 and 4.18).

ii. σ_z: variance of the error term ε_z in the instrument generation (4.16 and 4.18).

iii. d_u: number of unobserved confounders (which substitutes the percentage of causal confounders p_u not included in the nuisance functions in the backdoor adjustment scenario with wrong specification).

5 Orthogonal Score Function Derivation for Different Causal Scenarios

5.1 Orthogonal Score Function Derivation for the Backdoor Adjustment Scenario

In this appendix, we derive the orthogonal score function presented in equation (2.8) for the backdoor adjustment scenario Chernozhukov et al. (2018b); Pearl (2009).

We begin with the target parameter of interest, the Average Treatment Effect (ATE):

$$\tau = \mathbb{E}[Y(1) - Y(0)].$$

Under the Conditional Independence Assumption (CIA) and overlap, the ATE can be written as:

$$\tau = \mathbb{E}[\mu_1(X) - \mu_0(X)],$$

where $\mu_t(X) = \mathbb{E}[Y \mid T = t, X]$. Define the nuisance functions:

$$m(X) = \mathbb{E}[Y \mid X], \quad g(X) = \mathbb{E}[T \mid X].$$

We then express the observed data in terms of mean-zero residuals. Let:

$$\tilde{Y} := Y - m(X), \quad \tilde{T} := T - g(X).$$

By construction, if $m(\cdot)$ and $g(\cdot)$ are correctly specified, $\mathbb{E}[\tilde{Y} \mid X] = 0$ and $\mathbb{E}[\tilde{T} \mid X] = 0$.

Our goal is to derive a moment equation that identifies τ and is robust (orthogonal) to small perturbations in these nuisance functions. Consider the following moment condition:

$$\mathbb{E}\left[\frac{\tilde{T}}{g(X)(1-g(X))}\tilde{Y} + \mu_1(X) - \mu_0(X) - \tau\right] = 0.$$

Substituting $\mu_1(X) = \frac{\mathbb{E}[YT|X]}{g(X)}$ and $\mu_0(X) = \frac{\mathbb{E}[Y(1-T)|X]}{1-g(X)}$ and rearranging, we obtain the orthogonal score function:

$$\psi(Y, T, X; \tau, \eta) = \frac{T - g(X)}{g(X)(1-g(X))}(Y - m(X)) + [\mu_1(X) - \mu_0(X)] - \tau.$$

At the true parameter values and correctly specified nuisances, this score has expectation zero. Moreover, small first-order errors in $m(\cdot)$ or $g(\cdot)$ do not affect the leading-order bias in τ, ensuring Neyman-orthogonality.

5.2 Orthogonal Score Function Derivation for the Frontdoor Adjustment Scenario

In this appendix, we derive the orthogonal score function presented in equation (3.14) for the frontdoor adjustment scenario Chernozhukov et al. (2018b); Pearl (2009).

The frontdoor formula for the ATE is:

$$\tau = [h_{\text{FA}}(1) - h_{\text{FA}}(0)]\big(\mathbb{E}[Y \mid M = 1] - \mathbb{E}[Y \mid M = 0]\big),$$

where $h_{\text{FA}}(t) = \mathbb{E}[M \mid T = t]$ and $m_{\text{FA}}(m, c) = \mathbb{E}[Y \mid M = m, C = c]$.

Define the nuisance functions and residuals:

$$m_{\text{FA}}(M, C) = \mathbb{E}[Y \mid M, C], \quad h_{\text{FA}}(T) = \mathbb{E}[M \mid T].$$

Let:

$$\tilde{Y} := Y - m_{\text{FA}}(M, C), \quad \tilde{M} := M - h_{\text{FA}}(T).$$

By construction, if $m_{\text{FA}}(\cdot)$ and $h_{\text{FA}}(\cdot)$ are correct, $\mathbb{E}[\tilde{Y} \mid M, C] = 0$ and $\mathbb{E}[\tilde{M} \mid T] = 0$.

We seek a moment equation orthogonal to small perturbations in m_{FA} and h_{FA} that identifies τ. Consider:

$$\mathbb{E}\left[\frac{\tilde{M}\tilde{Y}}{h_{\text{FA}}(T)(1 - h_{\text{FA}}(T))}(h_{\text{FA}}(1) - h_{\text{FA}}(0)) - \tau\right] = 0.$$

Intuitively, $\frac{\tilde{M}\tilde{Y}}{h_{\text{FA}}(T)(1-h_{\text{FA}}(T))}$ recovers $\mathbb{E}[Y \mid M = 1] - \mathbb{E}[Y \mid M = 0]$ when the nuisances are correct, and multiplying by $(h_{\text{FA}}(1) - h_{\text{FA}}(0))$ yields τ.

This leads to the orthogonal score:

$$\psi(Y, T, M, C; \tau, \eta) = \left(\frac{M - h_{\text{FA}}(T)}{h_{\text{FA}}(T)(1 - h_{\text{FA}}(T))}(Y - m_{\text{FA}}(M, C))\right)(h_{\text{FA}}(1) - h_{\text{FA}}(0)) - \tau.$$

At the true parameter and correctly specified nuisances, this score has zero expectation and is orthogonal to first-order perturbations in m_{FA} and h_{FA}, ensuring Neyman-orthogonality.

5.3 Orthogonal Score Function Derivation for Instrumental Variable

In this appendix, we derive the orthogonal score function presented in equation (3.25) for the instrumental variable scenario Belloni et al. (2014); Chernozhukov et al. (2018b); Pearl (2009).

The main target parameter is the average treatment effect τ identified via an instrument Z. Define the nuisance functions:

$$m_{\mathrm{IV}}(C) = \mathbb{E}[Y \mid C], \quad g_{\mathrm{IV}}(C) = \mathbb{E}[T \mid C], \quad q_{\mathrm{IV}}(C) = \mathbb{E}[Z \mid C].$$

Subtracting the conditional means, we write:

$$\tilde{Y} := Y - m_{\mathrm{IV}}(C), \quad \tilde{T} := T - g_{\mathrm{IV}}(C), \quad \tilde{Z} := Z - q_{\mathrm{IV}}(C).$$

By construction, if the nuisance functions are correctly specified, $\mathbb{E}[\tilde{Y} \mid C] = 0$, $\mathbb{E}[\tilde{T} \mid C] = 0$, and $\mathbb{E}[\tilde{Z} \mid C] = 0$.

A suitable orthogonal moment condition for τ can be formed by considering a moment equation that is orthogonal with respect to perturbations in the nuisance functions:

$$\mathbb{E}[\tilde{Z}(\tilde{Y} - \tau\tilde{T})] = 0.$$

This moment equation identifies τ since it linearly relates $\tilde{Y}$ and $\tilde{T}$ with weight $\tilde{Z}$, which, by construction, is independent of the confounding that affects the relationship between T and Y.

Rewriting back in terms of the original variables and nuisance functions:

$$\mathbb{E}[(Z - q_{\mathrm{IV}}(C))((Y - m_{\mathrm{IV}}(C)) - \tau(T - g_{\mathrm{IV}}(C)))] = 0.$$

From this expectation equation, we derive the orthogonal score:

$$\psi(Y, T, Z, C; \tau, \eta) = (Z - q_{\mathrm{IV}}(C))\big((Y - m_{\mathrm{IV}}(C)) - \tau(T - g_{\mathrm{IV}}(C))\big).$$

By construction, $\mathbb{E}[\psi(Y, T, Z, C; \tau, \eta)] = 0$ at the true parameter τ and correct nuisance functions $\eta = \{m_{\mathrm{IV}}, g_{\mathrm{IV}}, q_{\mathrm{IV}}\}$. Furthermore, small perturbations in m_{IV}, g_{IV}, or q_{IV} only affect this expectation at second order, thus ensuring the Neyman-orthogonality property.

$$\psi(Y, T, Z, C; \tau, \eta) = (Z - q_{\mathrm{IV}}(C))\left[(Y - m_{\mathrm{IV}}(C)) - \tau(T - g_{\mathrm{IV}}(C))\right]. \tag{5.1}$$

6 Simulations

We build a simulation framework able for the outlined causal scenarios, tuning the parameters of the data generating process in accordance with the specifications outlined in Sections 4.3.5, 4.1.6, and 4.2.5.

The system is built using Python and SQLite, as detailed in the Appendix 8.2, and relies `DoubleML` package for the estimation of the causal effects. It works on generating by generating a simulation plan, in which each line refers to one scenario. Multiple processes read from the plan and run the simulations in parallel, storing the results in a SQLite database.

The scenarios use all the parameters specified in the Tuning section.

As a consequence of the use of cross-fitting and the generation of multiple scenarios, the simulation process is computationally intensive. [1]

The results are a sample of the multiple possibilies of DGP and provide interesting takeaways.

6.1 Simulation Plans

The parameters of the data generating process vary accordingly

- Treatment Noise Level: $\sigma_t = 0.1, 0.2, 0.5$
- Outcome Noise Level: $\sigma_y = 0.1, 0.2, 0.5$
- Number of Observations: $n = 100, 200, 1000, 5000, 10000$ (filter $n/d > 1.2$)
- Number of Causal Confounders: $d_c = 5, 10, 20, 50, 100$
- Number of Non-Causal Confounders: $d_a = 5, 10, 20, 50, 100$
- Alpha Sparsity: $\alpha = 0.3$
- Different relationships between treatment and target.
- Different relationships between covariates and treatment.

For each combination, we run 100 simulations, totaling 12 thousand. We run DML using the three different kinds of specification outlined: Correct, Inclusion of Non-Causal Confounders, and Partial Omission of Causal Cofounders with Inclusion of Non-Causal Confounders. For simplification, we run the equal propensity score and outcome models for DML. The chosen models are:

- LASSO (with polinomial features, cross-validation for regularization tuning)
- Elastic Net (with polinomial features, cross-validation for RIDGE and LASSO regularization tuning)
- OLS with Stepwise Selection (with polinomial features, and forward selection)
- Random Forest (with gridsearch for maximum depth and number of trees [2])
- Deep Neural Network (with gridsearch for layer architecture and regularization [3])

The combinations lead to 15 models for each simulation scenario, training 180 thousand models in total.

[1] In order to generate considerable amounts of results for Backdoor Adjustment, we ran for the duration of 5 days simulations in 4 different computers, each with 8-12 processes in parallel

[2] Number of trees gridsearch: {50, 100, 200, 500}; Maximum depth in gridsearch: $\{2, 3, 5\}$

[3] Layer architectures in gridsearch: {(32, 32, 32, 32, 32), (64, 64, 64, 64), (128, 64, 32), (32, 32, 32), (32, 32)}; Regularization in gridsearch: $\{0.1, 0.2, 0.5, 1\}$

6.2 Simulation Results

Specification	Inside CI
Correct	82.6%
Inclusion of Non-Causal Cofounders	76.6%
Unobserved Confounders	63.3%

Table 1: Percentage of Simulations where the true ATE is inside the 95% CI

d_c	Correct	Inclusion of Non-Causal Cofounders	Unobserved Confounders
5	0.062	0.098	0.127
10	0.248	0.387	0.503
20	1.548	2.338	3.039
50	4.494	6.803	8.843
100	21.060	30.083	39.108

Table 2: Mean Absolute Error per Specification and d_c

$n/(d_a + d_c)$	Correct	Inclusion of Non-Causal Cofounders	Unobserved Confounders
2	7.691172	9.596354	10.418657
5	6.664093	7.636725	7.880295
10	4.484867	5.363210	5.417331
12	0.863954	1.065748	1.138147
20	0.957901	1.104437	1.172468
25	0.507622	0.616798	0.629017
50	0.248625	0.288806	0.302507
100	0.081486	0.102542	0.108832
200	0.026309	0.033524	0.036599

Table 3: Mean Absolute Error per Specification and $n/(d_a + d_c)$

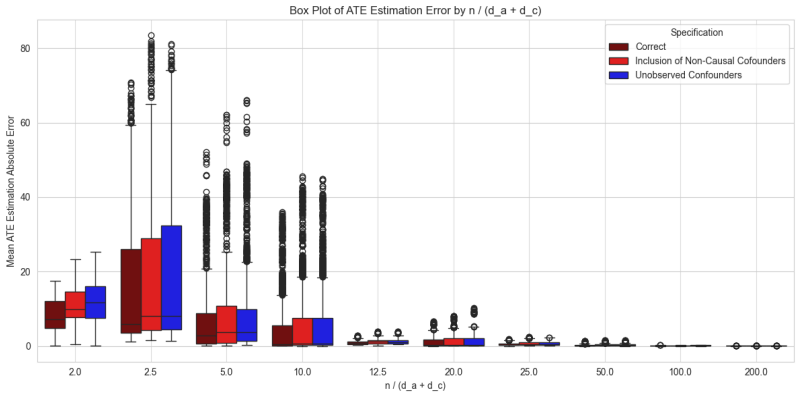


Figure 5: Boxplot of ATE Error by d_a (filtering samples in which $\frac{n}{d} > 10$)

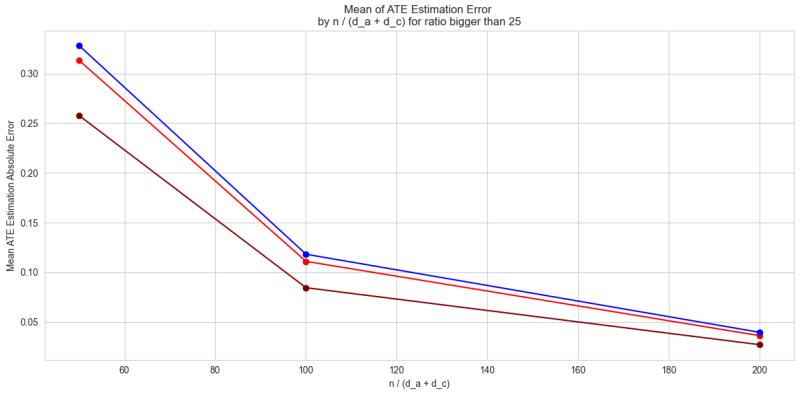


Figure 6: Boxplot of ATE Error by d_a (filtering samples in which $\frac{n}{d} > 10$)

Specification	Mean Absolute Error	Standard Deviaton of MAE
Correct	4.195484	8.312652
Inclusion of Non-Causal Cofounders	5.209074	8.589783
Unobserved Confounders	5.547834	8.679951

Table 4: Mean Absolute Error per Specification

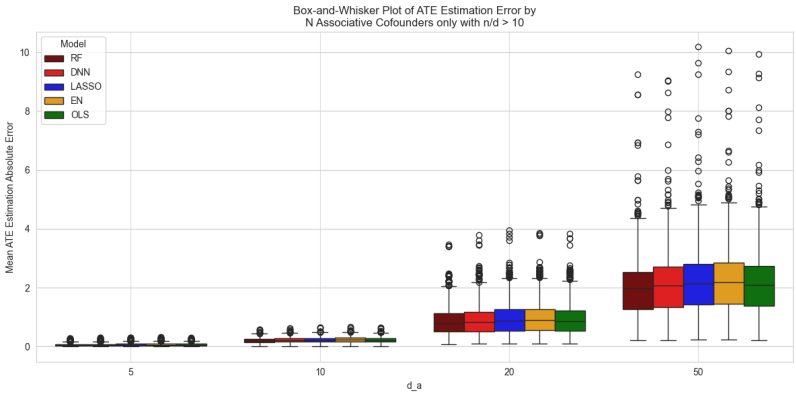


Figure 7: Boxplot of ATE Error by d_a (filtering samples in which $\frac{n}{d} > 10$)

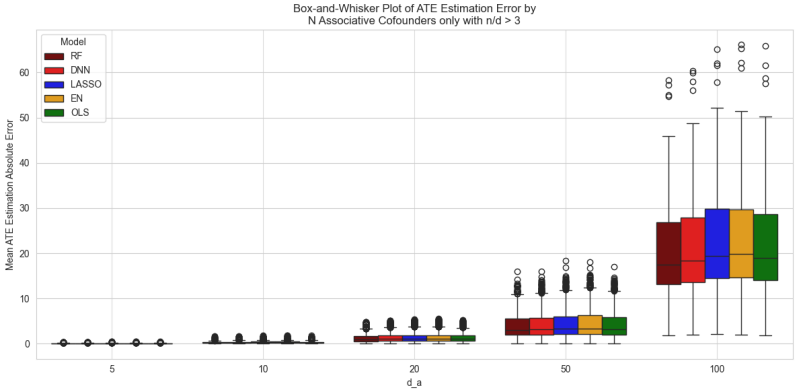


Figure 8: Boxplot of ATE Error by d_a (filtering samples in which $\frac{n}{d} > 3$)

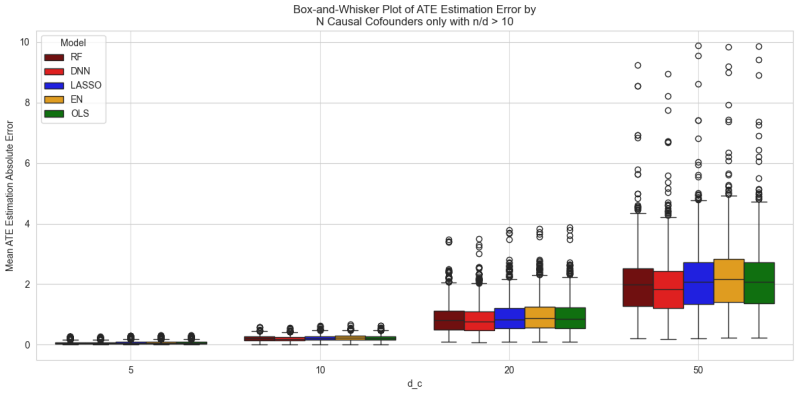


Figure 9: Boxplot of ATE Error by d_c (filtering samples in which $\frac{n}{d} > 10$)

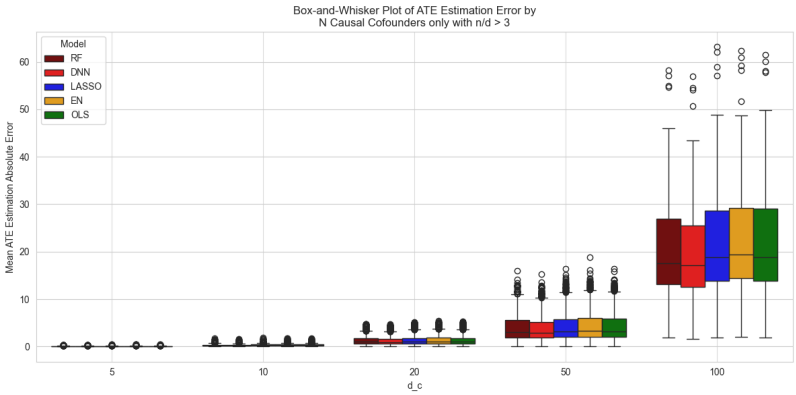


Figure 10: Boxplot of ATE Error by d_c (filtering samples in which $\frac{n}{d} > 3$)

7 Bibliography

References

Bang, H. & Robins, J. M. (2005). Doubly robust estimation in missing data and causal inference models. *Biometrics*, 61(4), 962–973.

Belloni, A., Chernozhukov, V., & Hansen, C. (2014). High-dimensional methods and inference on structural and treatment effects. *Journal of Economic Perspectives*, 28(2), 29–50.

Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., & Newey, W. (2018a). Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal*, 21(1), C1–C68.

Chernozhukov, V., Chetverikov, D., Demirer, P., Duflo, E., Hansen, C., & Newey, W. (2018b). Double machine learning for treatment and causal parameters. *arXiv preprint arXiv:1608.00060*.

Glynn, A. N. & Quinn, K. M. (2010). An introduction to the augmented inverse propensity weighted estimator. *Political Analysis*, 18(1), 36–56.

Imbens, G. W. & Rubin, D. B. (2015). *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. Cambridge, UK: Cambridge University Press.

Newey, W. K. (1990). Semiparametric efficiency bounds. *Journal of Applied Econometrics*, 5(2), 99–135.

Pearl, J. (1995). Causal diagrams for empirical research. *Biometrika*, 82(4), 669–688.

Pearl, J. (2009). *Causality: Models, Reasoning, and Inference*. Cambridge, UK: Cambridge University Press, 2nd edition.

Pearl, J. (2012). The front-door criterion for confounding control. In J. Pearl & M. Glymour (Eds.), *Causal Inference* chapter 4, (pp. 245–257). Cambridge, UK: Cambridge University Press.

Robins, J. M., Rotnitzky, A., & Zhao, L. P. (1994). Estimation of regression coefficients when some regressors are not always observed. *Journal of the American Statistical Association*, 89(427), 846–866.

Rubin, D. B. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5), 688–701.

8 Appendix

8.1 Mild Regularity Conditions for Double Machine Learning

i. Smoothness of the Target Function: The functions $m(X)$ and $g(X)$, representing the outcome and treatment models, should belong to a sufficiently smooth function class. This ensures that they can be approximated well by machine learning methods.

ii. Boundedness and Regularity: The outcome Y, treatment T, and covariates X should have bounded support or satisfy moment conditions, such as:

$$\mathbb{E}[|Y|^2] < \infty, \quad \mathbb{E}[|T|^2] < \infty.$$

Additionally, $m(X)$ and $g(X)$ should have bounded derivatives in some cases.

iii. Sparsity or Complexity Control: The nuisance estimators $\hat{m}(X)$ and $\hat{g}(X)$ should satisfy complexity restrictions, such as sparsity in high-dimensional settings or controlled VC dimensions, to ensure valid estimation and inference.

iv. Consistency and Rates of Convergence: The nuisance estimators $\hat{m}(X)$ and $\hat{g}(X)$ must converge to the true $m(X)$ and $g(X)$ at sufficiently fast rates (typically faster than $n^{-1/4}$ in terms of mean squared error).

v. Independence or Weak Dependence: Observations should be independent and identically distributed (i.i.d.) or satisfy weak dependence conditions, such as mixing properties for time-series data.

vi. Overlap Condition (Positivity): The propensity score $\pi(X) = \mathbb{P}(T = 1|X)$ must be bounded away from 0 and 1:

$$0 < c \leq \pi(X) \leq 1 - c \quad \text{for some } c > 0.$$

vii. Orthogonality of the Moment Function: The estimating equation or moment function used in DML should satisfy a doubly-robust property, meaning that small estimation errors in $\hat{m}(X)$ and $\hat{g}(X)$ do not affect the asymptotic distribution of the estimator.

viii. Cross-Fitting: Proper cross-fitting should be employed to ensure the orthogonality of the estimating equations and to avoid overfitting. This mitigates potential overfitting by using disjoint data splits for estimating nuisance parameters and evaluating the final moment function.

8.2 Computational System

An important part of the theoretical review and the simulation study is the computational system used to run the simulations. The system was built to provide robustness and reproducibility to the results. It is composed of the following components:

i. The database initialization and schema definition (`database` folder).

ii. The scenario classes that generate and simulate data: backdoor adjustment, frontdoor adjustment, and instrumental variable (`scenarios` folder).

iii. The tests and validation routines (`tests` folder).

iv. The simulation plans and automated running of multiple simulation configurations (`simulations` folder).

The code is entirely present in the following GitHub repository:

`https://github.com/Fernando-Urbano/causal-factor-investing`

8.2.1 Database Initialization and Schema Definition

`https://github.com/Fernando-Urbano/causal-factor-investing/database`

The database-related code initializes an SQLite database and defines a schema for storing simulation results.

```python
import sqlite3
import os

def load_sql(file_path):
    with open(file_path, 'r') as file:
        return file.read()

def initialize_database(db_path="database/causal_scenarios.db"):
    sql_query = load_sql("database/init/query_database_initialization.sql")
    with sqlite3.connect(db_path) as conn:
        cursor = conn.cursor()
        cursor.execute(sql_query)
```

Figure 11: Database Initialization

The schema allows for storing all the relevant information of the data generating process, results, and parameters of the simulations.

```
CREATE TABLE IF NOT EXISTS scenario_simulations (
    id INTEGER PRIMARY KEY AUTOINCREMENT
    , seed_data INTEGER
    , scenario_name TEXT
    , n_samples INTEGER
    , d_c INTEGER
    , d_a INTEGER
    , d_u INTEGER
    , timestamp_data_generation TEXT
    , noise_level_treatment REAL
    , noise_level_target REAL
    , noise_level_instrument REAL
    , noise_level_mediator REAL
    , alpha_corr_covariates REAL
    , true_ate REAL
    , avg_corr_covariates REAL
    , l_model TEXT
    , m_model TEXT
    , r_model TEXT
    , cv_loss_regression TEXT
    , cv_loss_classification TEXT
    , specification TEXT
    , pct_unobserved REAL
    , pct_extra_unobserved REAL
    , binary_instrument BOOLEAN
    , mediator_covariates_relationship_type TEXT
    , instrument_covariates_relationship_type TEXT
    , target_covariates_relationship_type TEXT
    , treatment_covariates_relationship_type TEXT
    , estimated_ate REAL
    , std_error REAL
    , ci_2_5_pct REAL
    , ci_97_5_pct REAL
    , estimated_ate_t_stat REAL
    , estimated_ate_p_value REAL
);
```

Figure 12: Database Initialization

8.2.2 Causal Scenario Classes

`https://github.com/Fernando-Urbano/causal-factor-investing/scenarios`

Under `scenarios/causal_scenarios.py`, we define classes that represent different causal settings: backdoor adjustment, frontdoor adjustment, and instrumental variables. Each class:

- Generates synthetic data with specified relationships and noise levels.
- Builds a DoubleML model to estimate the average treatment effect.
- Saves results into the database.

All the classes derive from a common abstract class `CausalScenario`, which defines the common interface for generating data, building models, and saving results.

```python

class CausalScenario:
    db_path = "database/causal_scenarios.db"

    def __init__(
            self,
            n_samples: int, d_c: int, d_a: int,
            noise_level_treatment: float,
            noise_level_target: float,
            alpha_corr_covariates: float,
            l_model: str = None,
            m_model: str = None,
            cv_loss_regression: str = CV_DEFAULT_LOSS_REGRESSION,
            cv_loss_classification: str = CV_DEFAULT_LOSS_CLASSIFICATION,
            target_covariates_relationship_type = "random",
            treatment_covariates_relationship_type = "random",
            specification: str = 'correct',
            constant_ite: bool = False,
            seed_data: int = 42,
        ):
        self._is_model_built = False
        if n_samples < MIN_N_SAMPLES:
            raise ValueError(f"n_samples must be at least {MIN_N_SAMPLES}")
        ....

class BackdoorAdjustmentScenario(CausalScenario):
    def generate_data(self):
        # Generate X, treatment, target according to a backdoor setup
        ...

    def build_model(self):
        # Create a DoubleMLPLR model for the scenario
        dml_data = DoubleMLData.from_arrays(x=X, y=self.target, d=self.treatment)
        self.dml = DoubleMLPLR(dml_data, ml_l=..., ml_m=..., n_folds=5)

class InstrumentalVariableScenario(CausalScenario):
    def generate_data(self):
        # Generate X, Z (instrument), treatment, and target for IV setup
        ...

    def build_model(self):
        # Create a DoubleMLIIVM model for IV scenarios
        dml_data = DoubleMLData.from_arrays(x=X, y=self.target, d=self.treatment, z=self.Z)
        self.dml = DoubleMLIIVM(dml_data, ml_g=..., ml_m=..., ml_r=..., n_folds=5)

class FrontdoorAdjustmentScenario(CausalScenario):
    def generate_data(self):
        ...
```

Figure 13: Causal Scenario Classes

Here we also define important functions, pipelines, and gridsearch options:

```python
NON_LINEAR_TRANSFORMATIONS = {
    "linear": lambda t: t,
    "sin": lambda t: np.sin(t),
    "cos": lambda t: np.cos(t),
    "square": lambda t: t**2,
    "cubic": lambda t: t**3,
    "step": lambda t: np.where(t > 0, 1, 0),
    "piecewise_linear": lambda t: np.piecewise(
        t, [t < 0, (t >= 0) & (t < 1), t >= 1],
        [lambda x: x * 2, lambda x: x, lambda x: x * 0.5]
    ),
    "exponential": lambda t: np.exp(t / 5),
    "logarithmic": lambda t: np.log1p(np.abs(t)),
}
```

Figure 14: Functions for Non-Linear Transformations

```
PARAM_GRID = {
    "RF": {
        'model__n_estimators': [100, 500],
        'model__max_depth': [2, 3, 5, 10]
    },
    "DNN": {
        'model__hidden_layer_sizes': [
            (32, 32, 32, 32, 32,),
            (64, 64, 64, 64),
            (128, 64, 32,),
            (32, 32, 32,),
            (32, 32,),
        ],
        'model__alpha': [.1, .2, .5, 1],  # Ridge regularization
    }
}
```

Figure 15: Gridsearch for DNN and RF

```
NORMAL_PIPELINES = {}
NORMAL_PIPELINES["classification"] {
    "LASSO": [
        ('poly_features', PolynomialFeatures(degree=2, include_bias=False)),
        ('scaler', StandardScaler()),
        ('model', LogisticRegressionCV(
            cv=5,
            penalty='l1',
            solver='saga',
            random_state=SEED_MODEL,
            max_iter=MAX_ITER_OLS
        ))
    ],
    "FS": [
        ('poly_features', PolynomialFeatures(degree=2, include_bias=False)),
        ('scaler', StandardScaler()),
        ('forward_selection', SequentialFeatureSelector(
            estimator=LogisticRegression(max_iter=MAX_ITER, random_state=SEED_MODEL),
            n_features_to_select='auto',
            direction='forward',
            scoring='accuracy',
            cv=5
        )),
        ('model', LogisticRegression(max_iter=MAX_ITER, random_state=SEED_MODEL))
    ],
    "EN": [
        ('poly_features', PolynomialFeatures(degree=2, include_bias=False)),
        ('scaler', StandardScaler()),
        ('model', LogisticRegressionCV(
            cv=5,
            penalty='elasticnet',
            solver='saga',
            l1_ratios=[0.25, 0.5, 0.75],
            random_state=SEED_MODEL,
            max_iter=MAX_ITER_OLS
        ))
    ],
    "RF": [('model', RandomForestClassifier(random_state=SEED_MODEL))],
    "DNN": [
        ('scaler', StandardScaler()),
        ('model', MLPClassifier(max_iter=MAX_ITER, random_state=SEED_MODEL, activation='relu', learning_rate='adaptive'))
    ]
},
```

Figure 16: Pipelines for Different Models in Classification

```
NORMAL_PIPELINES["regression"] =  {
    "LASSO": [
        ('poly_features', PolynomialFeatures(degree=2, include_bias=False)),
        ('scaler', StandardScaler()),
        ('model', LassoCV(cv=5, random_state=SEED_MODEL, max_iter=MAX_ITER_OLS))
    ],
    "FS": [
        ('poly_features', PolynomialFeatures(degree=2, include_bias=False)),
        ('scaler', StandardScaler()),
        ('forward_selection', SequentialFeatureSelector(
            estimator=LinearRegression(),
            n_features_to_select='auto',
            direction='forward',
            scoring='neg_mean_squared_error',
            cv=5
        )),
        ('model', LinearRegression())
    ],
    "EN": [
        ('poly_features', PolynomialFeatures(degree=2, include_bias=False)),
        ('scaler', StandardScaler()),
        ('model', ElasticNetCV(
            cv=5,
            l1_ratio=[0.25, 0.5, 0.75],
            random_state=SEED_MODEL,
            max_iter=MAX_ITER_OLS
        ))
    ],
    "RF": [
        ('model', RandomForestRegressor(random_state=SEED_MODEL))
    ],
    "DNN": [
        ('scaler', StandardScaler()),
        ('model', MLPRegressor(max_iter=MAX_ITER, random_state=SEED_MODEL, activation='relu', learning_rate='adaptive'))
    ]
}
```

Figure 17: Pipelines for Different Models in Regression

```
DATA_GENERATION_SPECIFICATION = {
    "BackdoorAdjustmentScenario": [
        "Correct",
        "Inclusion of Non-Causal Cofounders",
        "Unobserved Confounders"
    ],
    "InstrumentalVariableScenario": [
        "Correct",
        "Inclusion of Non-Causal Cofounders",
        "Extra Unobserved Confounders",
        "Instrument treated as Cofounder",
        "Instrument treated as Cofounder and Inclusion of Non-Causal Cofounders"
    ],
    "FrontdoorAdjustmentScenario": [
        "Correct",
        "Inclusion of Non-Causal Cofounders"
        "Ignoring the Mediator",
        "Considering the Mediator as a Normal Covariate"
    ]
}
```

Figure 18: Type of Data Specification or Misspecification for Different Causal Scenarios

8.2.3 Testing and Validation Routines

`https://github.com/Fernando-Urbano/causal-factor-investing/tests`

Under the `tests` folder, we have test scripts that:

i. Validate data generation under different specifications

ii. Check that the model fitting process runs successfully and produce the expected results.

iii. Ensure that the estimated ATE is close to the true ATE under controlled conditions.

iv. Validate non-linear relationships

```python
def test_backdoor_adjustment_correct_spec_no_noise():
    scenario = BackdoorAdjustmentScenario(
        n_samples=1000,
        d_c=1,
        d_a=3,
        noise_level_treatment=0.0,
        noise_level_target=0.0,
        alpha_corr_covariates=0.0,
        target_covariates_relationship_type = "linear",
        treatment_covariates_relationship_type = "linear",
        l_model='DNN',
        m_model='DNN',
        specification='Correct',
        seed_data=42
    )
    scenario.generate_data()
    scenario.build_model()
    scenario.fit_model()

    assert scenario.dml is not None
    estimated_ate = scenario.get_summary()['estimated_ate'].iloc[0]
    assert abs(estimated_ate - scenario.true_ate) < 0.1
```

Figure 19: Example of Test

We have 41 unittests which can all be run automatically using `pytest` in the terminal.

```
pytest -vv
============================================================ test session starts =============================================================

tests/test_generate_data.py::test_causal_scenario_generate_x PASSED                                                                  [  2%]
tests/test_generate_data.py::test_causal_scenario_generate_data PASSED                                                               [  5%]
...
```

Figure 20: Command to Run Unittests in Terminal

8.2.4 Simulation Plans and Automated Execution

`https://github.com/Fernando-Urbano/causal-factor-investing/simulations`

The `simulations` folder contains scripts that define plans to run simulations. Those plans define different parameters for each causal scenario, specification, noise level, relationship between variables, etc... The plan for each scenario is saved as a txt file and line by line read by the simulation script.

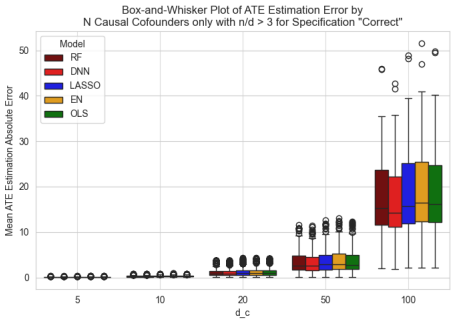
The code reads lines from text files, each specifying scenario parameters, runs the simulation, and saves the results. This process can be extended or parallelized to conduct comprehensive experiments. This allows us to run multiple process in parallel using the same computer (in which the optimal is roughly the number of CPUs available for performance).

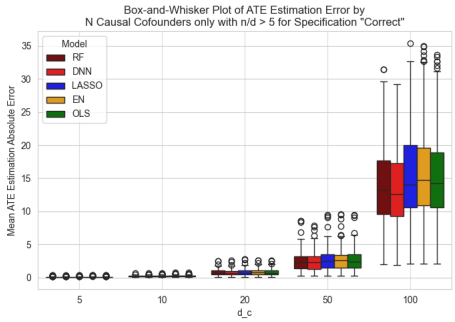
```
simulation_info = read_and_delete_second_line("simulations/plans/backdoor_simulation_plan.txt")
scenario = BackdoorAdjustmentScenario(
    n_samples=int(simulation_info['n_samples']),
    d_c=int(simulation_info['d_c']),
    ...
)
scenario.generate_data()
scenario.build_model()
scenario.fit_model()
scenario.save_summary()
```

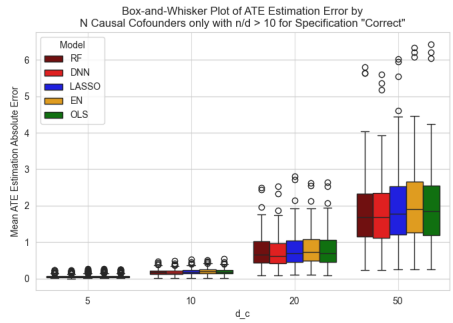
Figure 21: Automated Simulations Execution

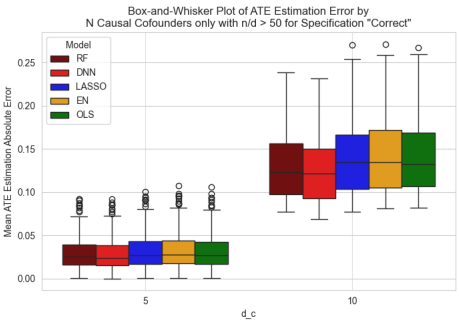
8.3 Other Simulations Results

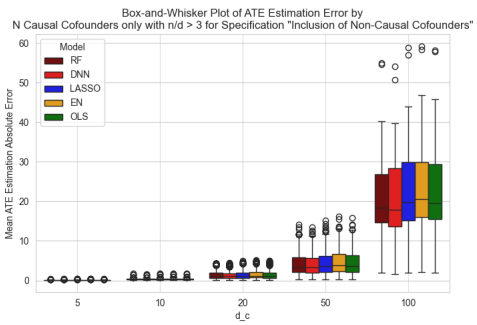
8.3.1 ATE Estimation Error by d_c and Specification

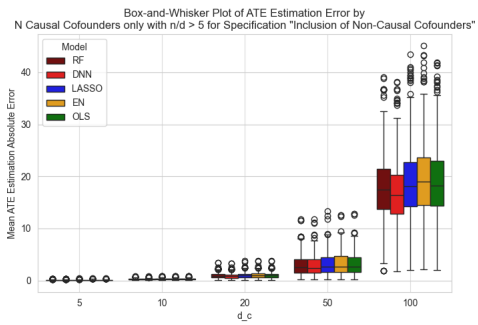


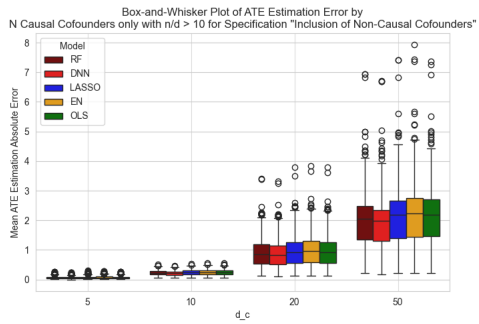


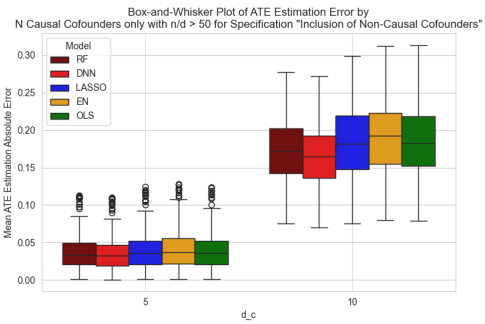


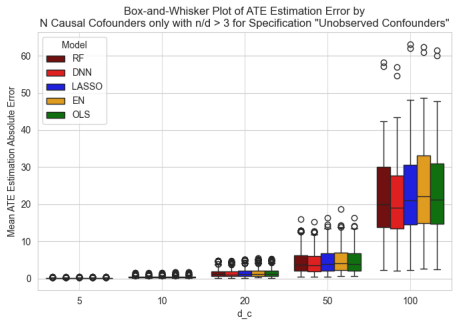


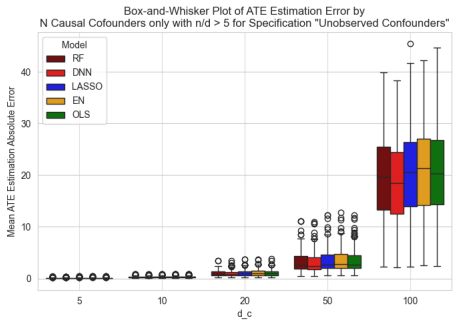


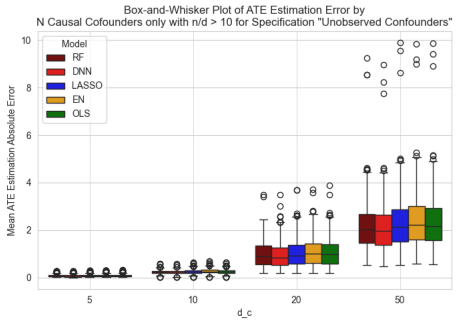


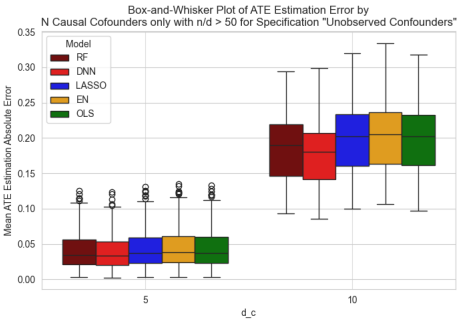




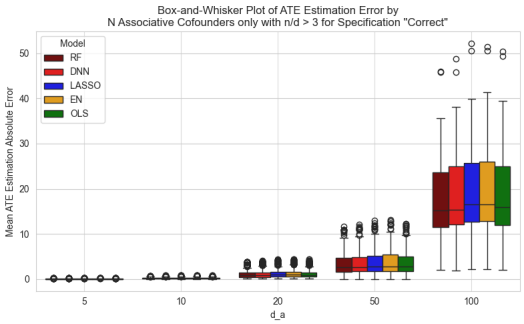


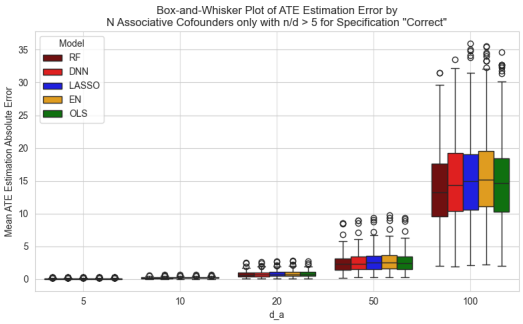


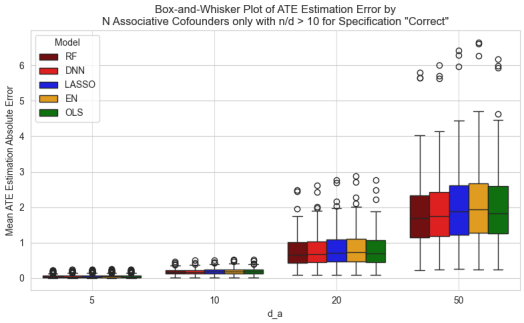


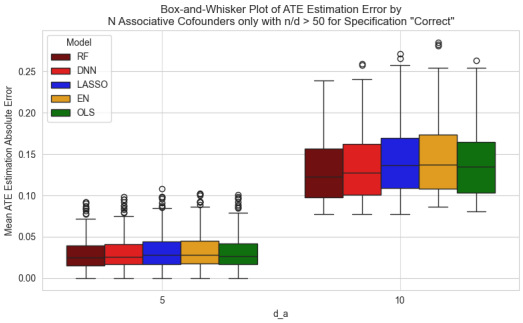


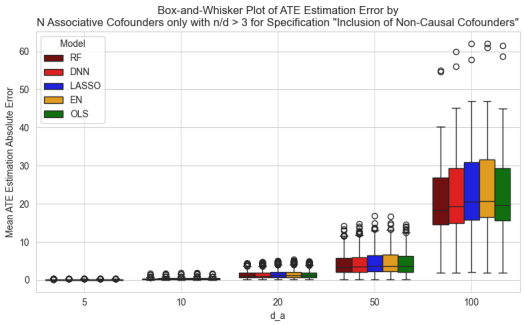
8.3.2 ATE Estimation Error by d_a and Specification

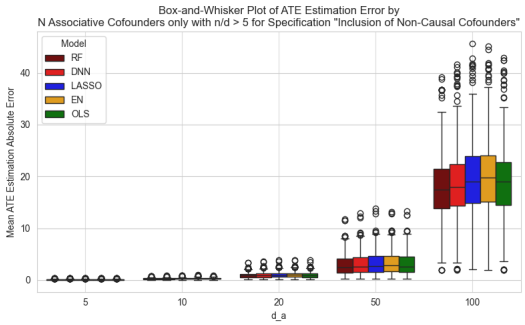


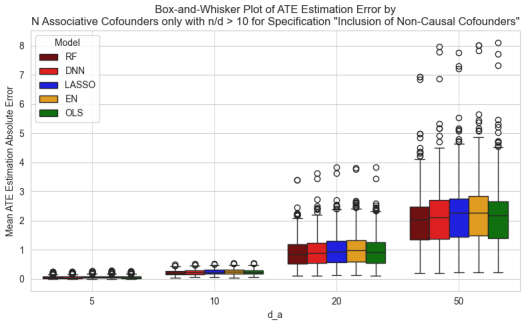


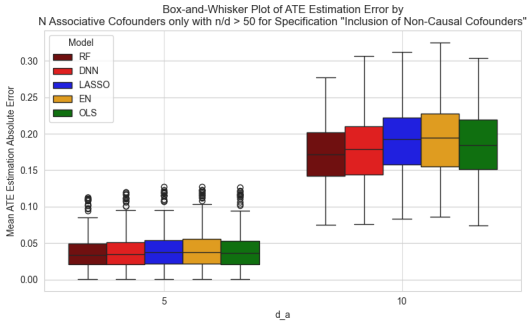


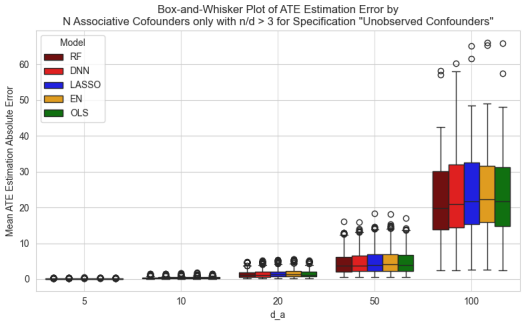


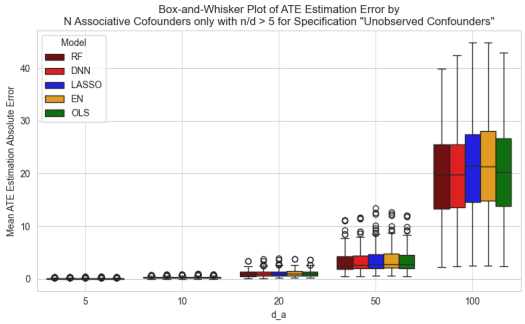

Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 3 for Specification "Inclusion of Non-Causal Cofounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a

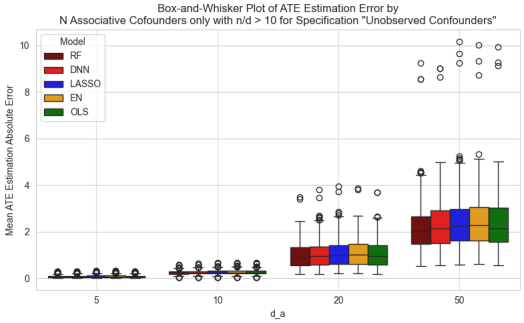

Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 5 for Specification "Inclusion of Non-Causal Cofounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a

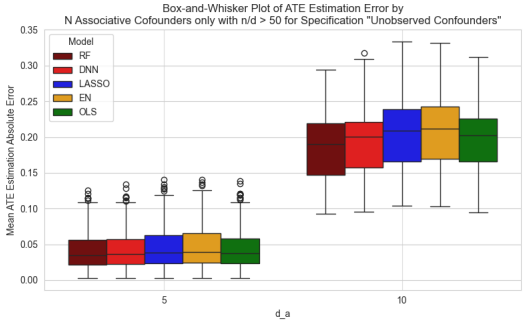

Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 10 for Specification "Inclusion of Non-Causal Cofounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a


Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 50 for Specification "Inclusion of Non-Causal Cofounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a


Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 3 for Specification "Unobserved Confounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a


Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 5 for Specification "Unobserved Confounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a


Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 10 for Specification "Unobserved Confounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a


Box-and-Whisker Plot of ATE Estimation Error by
N Associative Cofounders only with n/d > 50 for Specification "Unobserved Confounders"
Model
RF
DNN
LASSO
EN
OLS
Mean ATE Estimation Absolute Error
d_a