axioms

MDPI

Article

Treatment Effect Estimation in Survival Analysis Using Copula-Based Deep Learning Models for Causal Inference

Jong-Min Kim [1,2]

[1] Statistics Discipline, Division of Science and Mathematics, University of Minnesota-Morris, Morris, MN 56267, USA; jongmink@morris.umn.edu
[2] EGADE Business School, Tecnológico de Monterrey, Ave. Rufino Tamayo, Monterrey 66269, Mexico

Abstract: This paper presents the use of Copula-based deep learning with Horvitz–Thompson (HT) weights and inverse probability of treatment weighting (IPTW) for estimating propensity scores in causal inference problems. This study compares the performance of the statistical methods—Copula-based deep learning with HT and IPTW weights, propensity score matching (PSM), and logistic regression—in estimating the treatment effect (ATE) using both simulated and real-world data. Our results show that the Copula-based recurrent neural network (RNN) with the method of HT weights provides the most precise and robust treatment effect estimate, with narrow confidence intervals indicating high confidence in the results. The PSM model yields the largest treatment effect estimate, but with greater uncertainty, suggesting sensitivity to data imbalances. In contrast, logistic regression and causal forests produce a substantially smaller estimate, potentially underestimating the treatment effect, particularly in structured datasets such as COMPAS scores. Overall, copula-based methods (HT and IPTW) tend to produce higher and more precise estimates, making them effective choices for treatment effect estimation in complex settings. Our findings emphasize the importance of method selection based on both the magnitude and precision of the treatment effect for accurate analysis.

Keywords: causal inference; deep learning; copula; survival analysis

MSC: 62-08

check for updates

Academic Editor: Oscar Montiel Ross

Received: 22 April 2025
Revised: 30 May 2025
Accepted: 9 June 2025
Published: 10 June 2025

Citation: Kim, J.-M. Treatment Effect Estimation in Survival Analysis Using Copula-Based Deep Learning Models for Causal Inference. *Axioms* **2025**, *14*, 458. https://doi.org/10.3390/axioms14060458

Copyright: © 2025 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Estimating treatment effects is a fundamental task in observational studies, especially when randomized controlled trials (RCTs) are not feasible. Accurate estimation of the average treatment effect (ATE) is critical in fields such as economics, healthcare, and the social sciences, where understanding the impact of interventions informs policy and practice. However, observational data often present challenges such as confounding, treatment selection bias, and complex interdependencies among covariates [1,2].

Traditional approaches, such as PSM and logistic regression, are widely used to address these issues. PSM attempts to reduce bias by matching treated and control units based on the probability of treatment assignment, while logistic regression directly models treatment assignment as a function of covariates [1,3]. Despite their utility, these methods have notable limitations. PSM may struggle with unmeasured confounding and imbalance across groups [4], while logistic regression relies on strong parametric assumptions that may not hold in complex settings [5].

When estimating heterogeneous treatment effects (HTEs), particularly in survival analysis, the limitations of classical models become even more pronounced [6]. The Cox proportional hazards model, for example, assumes a constant treatment effect across populations, which may obscure meaningful subgroup differences [7]. Moreover, parametric and semiparametric models often impose assumptions such as proportional hazards, which are frequently violated in real-world data [8]. Machine learning methods, such as causal forests [9], have been adapted for HTE estimation but still face difficulties in handling censoring without relying on strong assumptions.

Recent advancements in survival modeling include DeepSurv, a deep neural network version of the Cox model that learns complex interactions between covariates and treatment [10]. Simulation studies, such as those by [11], have evaluated the performance of machine learning methods under various levels of confounding and covariate overlap. Other approaches, like the dynamic survival models proposed by [12], utilize longitudinal data and deep learning for improved predictive accuracy.

Beyond survival analysis, innovations in machine learning continue to expand. Studies include motion prediction in dynamic systems [13], fast detection methods in power grids [14], planning architectures with memory-integrated language models [15], and enhanced SQL parsing for electronic medical records [16]. Additionally, bioanalytical and clinical studies have explored pharmacokinetic quantification [17], neuroprotective agents [18], immune-related adverse effects [19], and molecular mechanisms in cancer [20]. These diverse studies reflect a broader trend of leveraging computational models to understand complex data and improve decision-making.

Recently, Copula-based models have emerged as powerful tools to capture nonlinear and non-Gaussian dependencies among variables [21,22]. Copulas decouple marginal distributions from dependence structures, offering a flexible framework to model relationships between covariates, treatment assignments, and outcomes. Copula methods also show promise in handling unobserved confounding and missing data.

In this study, we propose and evaluate a Copula RNN with the HT weights framework, which integrates Copula-based transformations with RNNs to capture dynamic, nonlinear dependencies and adjust for treatment selection bias. We compare this method with PSM, causal forests, and logistic regression using both simulated and real-world datasets.

By analyzing the accuracy, robustness, and interpretability of treatment effect estimates across these approaches, this work provides practical guidance for selecting appropriate causal inference methods in the presence of complex dependencies. This study contributes to the growing literature on data-driven causal analysis and enhances methodological options for treatment effect estimation in high-dimensional, observational contexts.

2. Experimental Details

To evaluate treatment effect estimation methods, we generate synthetic data with n samples and p features. Each sample i has covariates $\mathbf{X}_i = (X_{i1}, X_{i2}, \ldots, X_{ip})^\top$ drawn from a multivariate normal distribution:

$$\mathbf{X}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$$

where the covariance matrix $\mathbf{\Sigma}$ is structured as

$$\Sigma_{jk} = \begin{cases} 1, & \text{if } j = k \quad \text{(variance)} \\ \rho, & \text{if } j \neq k \quad \text{(correlation, set to 0.8)} \end{cases}$$

ensuring strong correlations among features.

To introduce nonlinear dependencies, we apply the following transformations:

$$X_{101} = \sin(X_1) \cdot \cos(X_2)$$

$$X_{102} = \log(|X_3| + 1) \cdot e^{-X_4}$$

$$X_{103} = X_5^2 + X_6^3$$

These transformations introduce higher-order interactions, enhancing the complexity of the feature space.

We include key variables in the simulation setup as follows: we denote T_i as the true survival time for individual i, C_i as the censoring time, and $T_i^* = \min(T_i, C_i)$ as the observed survival time. The event indicator δ_i is defined as

$$\delta_i = \begin{cases} 1, & \text{if the event occurred} \\ 0, & \text{if the observation is censored} \end{cases}$$

Survival times T_i follow a Weibull distribution:

$$T_i \sim \text{Weibull}(\lambda_i, k)$$

where the shape parameter is $k = 2$, and the scale parameter λ_i is defined as

$$\lambda_i = \exp(\beta_1 X_{i1} + \beta_2 X_{i2})$$

Censoring times C_i are drawn from a uniform distribution:

$$C_i \sim \text{Uniform}(0, \alpha \max(T))$$

where $\alpha = 0.8$. The event indicator δ_i is defined as

$$\delta_i = \begin{cases} 1, & \text{if } T_i \leq C_i \\ 0, & \text{otherwise} \end{cases}$$

3. Statistical Methods

3.1. Causal Weighting Methods in Survival Analysis

Survival analysis plays a critical role in fields such as healthcare, finance, and engineering, where modeling time-to-event outcomes is essential. Machine learning models have significantly improved predictive accuracy for such outcomes. However, in observational settings where treatment assignment is not randomized, causal inference methods are necessary to address confounding. This study evaluates survival models augmented with HT weights and IPTW to enable causal interpretation. HT weights originate from survey sampling and are adapted to causal inference [23]. IPTW reweights the sample to construct a pseudo-population in which treatment is independent of covariates [23].

Let $Z_i \in \{0, 1\}$ denote the binary treatment indicator, Y_i the observed outcome (e.g., time-to-event outcome), and $\mathbf{X}_i \in \mathbb{R}^p$ the vector of pre-treatment covariates for individual i. The propensity score is defined as

$$e(\mathbf{X}_i) = P(Z_i = 1 \mid \mathbf{X}_i),$$

representing the conditional probability of receiving treatment.

HT weights originate from survey sampling and are adapted to causal inference to correct for unequal probabilities of treatment assignment. IPTW similarly reweights the sample to construct a pseudo-population in which treatment assignment is independent of observed covariates. Though they have different origins, the functional form of the weights and the resulting ATE estimators are mathematically equivalent in this context.

The weight assigned to each unit is

$$w_i = \begin{cases} \frac{1}{e(\mathbf{X}_i)} & \text{if } Z_i = 1, \\ \frac{1}{1-e(\mathbf{X}_i)} & \text{if } Z_i = 0, \end{cases}$$

where $e(\mathbf{X}_i) = P(Z_i = 1 \mid \mathbf{X}_i)$ is the propensity score.

The resulting estimator for the average treatment effect (ATE) is

$$\widehat{\text{ATE}} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{Z_i Y_i}{e(\mathbf{X}_i)} - \frac{(1 - Z_i) Y_i}{1 - e(\mathbf{X}_i)} \right).$$

This formulation is referred to as the Horvitz–Thompson (HT) estimator in the survey sampling literature, and as the IPTW estimator in the causal inference literature. Both estimators are unbiased under the assumptions of no unmeasured confounding and positivity [23].

To improve numerical stability, stabilized weights can be used:

$$w_i^{\text{stab}} = \frac{P(Z_i)}{P(Z_i \mid \mathbf{X}_i)}.$$

3.2. Logistic Regression for Propensity Score Estimation

Causal inference methods aim to estimate treatment effects from observational data, where treatment is not randomly assigned but may be influenced by individual characteristics, introducing potential confounding and selection bias. Propensity score methods address confounding by balancing covariates across treatment groups [2]. Matching on the propensity score helps reduce confounding bias [4]. This study employs logistic regression to estimate propensity scores, which are subsequently applied in IPTW, HT weights, and PSM.

Given n individuals and d covariates, the covariate matrix is defined as

$$\mathbf{X} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1d} \\ X_{21} & X_{22} & \cdots & X_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{nd} \end{bmatrix}. \tag{1}$$

Let $Z_i \in \{0, 1\}$ denote the binary treatment assignment for an individual i, where $Z_i = 1$ indicates the receipt of the treatment and $Z_i = 0$ indicates control. The probability of receiving the treatment, given the covariates $\mathbf{X}_i$, is modeled using the logistic function

$$e(\mathbf{X}_i) = P(Z_i = 1 \mid \mathbf{X}_i) = \frac{\exp(\mathbf{X}_i^\top \boldsymbol{\beta})}{1 + \exp(\mathbf{X}_i^\top \boldsymbol{\beta})}, \tag{2}$$

where $\boldsymbol{\beta} \in \mathbb{R}^d$ is the coefficient vector.

The parameter $\boldsymbol{\beta}$ is estimated via maximum likelihood by maximizing the binomial log-likelihood:

$$\log L = \sum_{i=1}^{n} [Z_i \log e(\mathbf{X}_i) + (1 - Z_i) \log(1 - e(\mathbf{X}_i))]. \quad (3)$$

By modeling treatment assignment probabilistically, logistic regression enables the estimation of propensity scores in binary treatment settings. These scores are crucial for causal inference and are incorporated into IPTW, HT weighting, and matching techniques to adjust for confounding and estimate treatment effects.

3.3. Causal Forests

Causal forests are a machine learning approach specifically designed to estimate heterogeneous treatment effects (HTEs), which refer to variations in treatment effects across different individuals or subpopulations within a dataset. Unlike traditional methods that focus on estimating an average treatment effect (ATE) for the entire population, causal forests aim to capture how the effect of a treatment or intervention changes based on covariates, thereby providing conditional average treatment effect (CATE) estimates.

Built as an extension of random forests, causal forests leverage the flexibility and non-parametric nature of decision tree ensembles to model complex, nonlinear relationships and interactions among variables without requiring explicit specification of functional forms. They modify the tree-growing process to prioritize splits that maximize heterogeneity in treatment effects rather than merely optimizing prediction accuracy.

A key feature of causal forests is the use of honest trees, which employ sample splitting to reduce overfitting and improve inference validity. Specifically, the data are partitioned into two independent subsets: one subset is used to determine the structure of the trees (i.e., where to split), and the other subset is used to estimate treatment effects within the leaves. This separation ensures that treatment effect estimates are unbiased and enables valid confidence interval construction.

The algorithm proceeds as follows: The dataset is randomly divided into two parts for honest estimation. Trees are grown by recursively splitting the data to maximize differences in treatment effects between resulting child nodes. Within each leaf node, the treatment effect is estimated as the difference in mean outcomes between treated and control units. Multiple trees are aggregated to form an ensemble, which smooths the estimates, reduces variance, and enhances robustness.

Causal forests have been widely applied in fields such as healthcare for personalized treatment recommendations, marketing for customer targeting, and policy evaluation to identify subpopulations most impacted by interventions [9,24].

3.4. Gaussian Copula Transformation

The Gaussian Copula is a flexible tool for modeling the dependence structure among random variables while preserving their marginal distributions. This section introduces the Gaussian Copula transformation, its mathematical formulation, and its application in data modeling. We employed the Gaussian Copula due to its computational convenience and reasonable performance when dependencies are symmetric and moderate. However, it is limited in capturing tail dependence or asymmetric relationships.

A Copula is a multivariate distribution function that connects univariate marginals to a joint distribution. By Sklar's theorem [25], any multivariate joint distribution $F_{\mathbf{X}}(x_1, \ldots, x_d)$ can be decomposed into its marginals $F_{X_i}(x_i)$ and a Copula C that captures the dependency structure:

$$F_{\mathbf{X}}(x_1, x_2, \ldots, x_d) = C\big(F_{X_1}(x_1), F_{X_2}(x_2), \ldots, F_{X_d}(x_d)\big), \tag{4}$$

where $C : [0,1]^d \to [0,1]$ is the Copula function and F_{X_i} are the marginal CDFs of X_i.

The Gaussian Copula is derived from the multivariate normal distribution. Let $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)$ be a multivariate normal vector with mean zero and correlation matrix Σ:

$$\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, \Sigma). \tag{5}$$

Then, the Gaussian Copula C_Σ is defined as

$$C_\Sigma(u_1, u_2, \ldots, u_d) = \Phi_\Sigma\Big(\Phi^{-1}(u_1), \Phi^{-1}(u_2), \ldots, \Phi^{-1}(u_d)\Big), \tag{6}$$

where

- Φ is the standard normal CDF;
- Φ^{-1} is its quantile function;
- Φ_Σ is the joint CDF of a multivariate normal distribution with correlation matrix Σ.

This construction separates the marginal behavior from the dependence structure.

The Gaussian Copula transformation offers a key advantage over standard scaling methods (e.g., min–max scaling) used in many deep learning applications. While standard scaling simply rescales each variable independently to a $[0,1]$ range, it does not account for the dependency structure among variables.

In contrast, the Gaussian Copula transformation maps multivariate data to a multivariate uniform distribution on $[0,1]^n$ while preserving the dependence structure between variables. This ensures that no information about the inter-variable relationships is lost. This dependency-aware transformation can be particularly beneficial when modeling complex relationships in deep learning, especially for tasks where interactions between features are important.

3.4.1. Transforming Data for Copula Modeling

To apply a Copula model, each variable must first be transformed to a uniform distribution on $[0,1]$. This is achieved using either the empirical CDF (ECDF) or parametric CDFs. For the ECDF, the transformation for each observation x_{ij} of variable X_j is

$$u_{ij} = \widehat{F}_{X_j}(x_{ij}), \tag{7}$$

where $\widehat{F}_{X_j}$ is the empirical CDF of X_j, and $u_{ij} \sim \text{Uniform}(0,1)$. The transformed data $\mathbf{u}_i = (u_{i1}, u_{i2}, \ldots, u_{id})$ are called pseudo-observations.

3.4.2. Parameter Estimation via Maximum Likelihood

The dependency structure of the Gaussian Copula is fully specified by its correlation matrix Σ. To estimate Σ, the log-likelihood of the Gaussian Copula given pseudo-observations $\{\mathbf{u}_i\}_{i=1}^n$ is maximized:

$$\hat{\Sigma} = \arg\max_{\Sigma} \sum_{i=1}^{n} \log c_\Sigma(\mathbf{u}_i), \tag{8}$$

where $c_\Sigma(\mathbf{u}_i)$ is the Copula density:

$$c_\Sigma(\mathbf{u}_i) = \frac{\phi_\Sigma(\Phi^{-1}(u_{i1}), \ldots, \Phi^{-1}(u_{id}))}{\prod_{j=1}^{d} \phi(\Phi^{-1}(u_{ij}))}, \tag{9}$$

with ϕ_Σ the PDF of the multivariate normal distribution and ϕ the standard normal density.

3.4.3. Simulation from the Gaussian Copula

Once $\hat{\Sigma}$ is estimated, new samples preserving the dependence structure can be generated as follows:

1. Simulate $\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, \hat{\Sigma})$.
2. Transform to uniform marginals: $u_{ij} = \Phi(z_{ij})$.
3. Map to the original scale using inverse marginal CDFs:

$$x_{ij} = F_{X_j}^{-1}(u_{ij}), \quad \forall j = 1, \ldots, d. \tag{10}$$

This procedure ensures that the synthetic data match both the empirical marginals and the estimated joint dependence structure.

3.4.4. Applications in Survival Analysis and Causal Inference

Gaussian Copula transformations have proven useful in modeling complex dependencies between high-dimensional covariates:

- Survival Analysis: Copula transformations allow for modeling correlated time-to-event features, improving calibration and risk stratification.
- Causal Inference: Improved balance in covariates via Copula-based propensity score modeling leads to more accurate treatment effect estimation.
- Deep Learning Pipelines: Copula-preprocessed features improve training stability and performance in LSTM-based causal models.

By decoupling marginal behavior from dependence structure, Copula models generate realistic synthetic data, enhance model interpretability, and facilitate robust statistical inference.

3.5. Bootstrapped Confidence Intervals

Bootstrapping is a nonparametric resampling method used to estimate the variability of a statistic by repeatedly drawing samples with replacement from the observed data. It is particularly useful in survival analysis and causal inference, where analytical variance estimators may be complex or intractable.

Let $\hat{\theta}$ be the point estimate of the parameter of interest (e.g., treatment effect or survival probability). The bootstrap procedure involves the following steps:

1. Resample the observed dataset $D = \{(X_i, Y_i, A_i, \ldots)\}_{i=1}^n$ to generate a bootstrap sample D_b^* of size n, sampled with replacement.
2. Compute the statistic of interest $\hat{\theta}_b^*$ using D_b^*.
3. Repeat steps 1 and 2 for B bootstrap iterations, obtaining bootstrap estimates $\{\hat{\theta}_1^*, \hat{\theta}_2^*, \ldots, \hat{\theta}_B^*\}$.
4. Construct the confidence interval using empirical quantiles of the bootstrap distribution.

3.5.1. Types of Bootstrapped Confidence Intervals

Several methods exist for constructing bootstrapped confidence intervals (BCIs):

Percentile Method

The confidence interval is estimated using the empirical quantiles of the bootstrap distribution:

$$\left(\hat{\theta}_{(\alpha/2)}^*, \hat{\theta}_{(1-\alpha/2)}^*\right), \tag{11}$$

where $\hat{\theta}^*_{(q)}$ denotes the q-th quantile of the bootstrap estimates and α is the significance level (e.g., 0.05 for a 95% CI).

Bias-Corrected and Accelerated (BCa) Method

This method adjusts for bias and skewness in the bootstrap distribution:

$$\left(\hat{\theta}^*_{\Phi\left(z_0 + \frac{z_0 + z_{\alpha/2}}{1 - a(z_0 + z_{\alpha/2})}\right)}, \quad \hat{\theta}^*_{\Phi\left(z_0 + \frac{z_0 + z_{1-\alpha/2}}{1 - a(z_0 + z_{1-\alpha/2})}\right)} \right), \tag{12}$$

where

- z_0 is the bias-correction factor;
- a is the acceleration parameter;
- $z_{\alpha/2}$ is the $\alpha/2$-th quantile of the standard normal distribution;
- $\Phi(\cdot)$ is the standard normal CDF.

Normal Approximation Method

Assuming approximate normality of the estimator, the confidence interval is

$$\hat{\theta} \pm z_{\alpha/2} \cdot \hat{\sigma}^*, \tag{13}$$

where $\hat{\sigma}^*$ is the standard deviation of the bootstrap estimates.

In our analysis, we used the percentile bootstrap confidence interval, which is a commonly recommended method due to its simplicity and minimal assumptions. Specifically, we generated a large number of bootstrap resamples (with replacement) of the original data and computed the statistic of interest (e.g., the mean) for each resample. The confidence interval was then constructed by taking the appropriate lower and upper quantiles (e.g., 2.5% and 97.5% for a 95% confidence level) from the distribution of the bootstrap statistics.

This method has the advantage of being entirely nonparametric, requiring no assumptions about the shape of the sampling distribution, and is particularly effective when the underlying distribution is unknown or asymmetric.

3.5.2. Bootstrapping in Causal Inference and Survival Analysis

Bootstrapping, introduced by [26], is a resampling technique used to quantify the uncertainty of statistical estimators by repeatedly drawing samples with replacement from the original dataset. In causal inference, bootstrapping is widely employed to assess the sampling variability of the average treatment effect (ATE).

Given individual treatment effect estimates $\hat{\tau}_i$ for treated units i such that $T_i = 1$, the ATE is computed as

$$\widehat{\text{ATE}} = \frac{1}{n_T} \sum_{i:T_i=1} \hat{\tau}_i, \tag{14}$$

where $n_T = \sum_{i=1}^{n} \mathbb{I}(T_i = 1)$ is the number of treated subjects.

By generating multiple bootstrap samples, recalculating propensity scores, updating matches or weights, and re-estimating the ATE for each resample, one obtains an empirical distribution of ATE estimates. This distribution is used to construct confidence intervals, quantify estimation uncertainty, and evaluate the statistical significance of the treatment effect.

In survival analysis, bootstrapping is also applied to estimate the variability of Kaplan–Meier survival curves, hazard ratios, and model-based survival predictions (e.g., from Cox models or deep learning-based survival models). The bootstrap distribution of survival

metrics enables the construction of confidence intervals that reflect the reliability of survival estimates. Narrow intervals indicate higher precision, while wider intervals signal greater uncertainty. Notably, if the confidence interval for the ATE includes zero, it suggests that the estimated treatment effect is not statistically significant at the chosen confidence level.

3.6. Sensitivity Analysis

Sensitivity analysis evaluates the robustness of causal inference results by testing their stability under variations in model assumptions or input data. In this study, sensitivity analysis is performed on the following models: Copula RNN with HT weights, Copula RNN with IPTW, PSM, logistic regression, and causal forests. Each model is evaluated under two conditions: (i) the original covariate data, and (ii) perturbed covariates obtained by adding 10% Gaussian noise. This perturbation assesses the resilience of model performance and treatment effect estimates to moderate changes in the data distribution, providing insight into model robustness.

4. Main Results

4.1. Simulated Data Analysis

We generated highly correlated simulated data to evaluate the models. A random seed was set to ensure reproducibility. The number of samples (n) was set to 1000, and the number of features (p) to 6. A correlation matrix $\mathbf{\Sigma} \in \mathbb{R}^{p \times p}$ was constructed with pairwise correlations of 0.8 off the diagonal and 1 on the diagonal.

Synthetic data $\mathbf{X} = (X_1, X_2, \ldots, X_p)$ with the specified correlation structure were generated using the multivariate normal distribution:

$$\mathbf{X}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \quad i = 1, \ldots, n.$$

To introduce nonlinear relationships, transformations such as sine, cosine, logarithm, and exponentiation were applied to the original features $X_1, \ldots, X_6$.

Survival times T_i for each observation were sampled from a Weibull distribution:

$$T_i \sim \text{Weibull}(\lambda_i, k), \tag{15}$$

where the scale parameter λ_i is defined as a function of features X_1 and X_2, and $k > 0$ is the shape parameter. Event types (1 and 2) were randomly assigned, and censoring times were simulated from a uniform distribution. The observed times were set as the minimum of survival and censoring times, with the event indicator taking values 0 (censored) or $1/2$ (event occurred).

A logistic regression model was used to estimate treatment probabilities (propensity scores), representing the likelihood of each event type given the features. The inverse of these propensity scores was used to compute IPTW and HT weights.

A Gaussian Copula transformation was applied to model the dependency structure among features. The Copula model was estimated using maximum likelihood estimation (`fitCopula`), and transformed features were generated using `rCopula`.

4.1.1. LSTM Model Training

An LSTM model was implemented using the `keras` package. The architecture consisted of an LSTM layer followed by a dense output layer. The model was trained using the MSE loss function and Adam optimizer, with sample weights set to HT weights. A second LSTM model was trained using IPTW weights.

To provide an additional treatment effect estimate, PSM was performed using the `matchit` package. Treatment groups were matched based on features, and the treatment effect estimate was calculated as the average survival time difference between matched pairs.

4.1.2. Bootstrapped Confidence Intervals

A function `bootstrap_ci` was defined to compute confidence intervals (CIs) for treatment effect estimates using resampling. For each bootstrap sample, predictions were drawn from the trained models, and the mean was calculated. The empirical quantiles of the bootstrap distribution were used to construct confidence intervals in Figure 1.

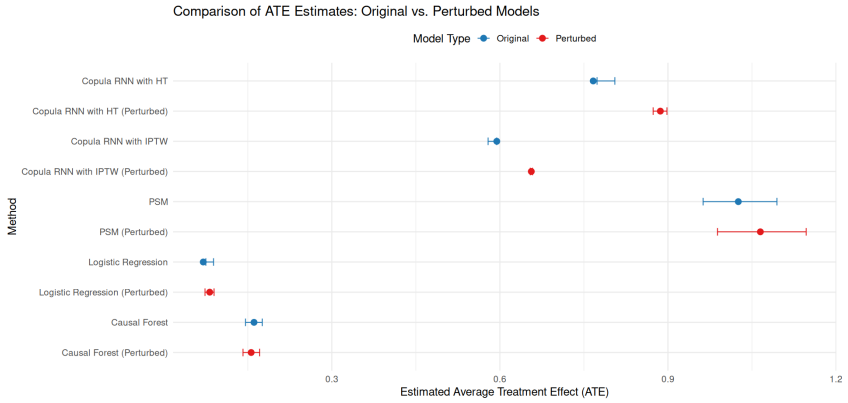


Figure 1. Treatment Effect Estimates with Confidence Intervals (Original vs. Perturbed Models) for Simulated Data.

Table 1 presents the estimated Average Treatment Effects (ATEs) and their corresponding 95% confidence intervals (CIs) across five causal inference methods, evaluated under both the original and perturbed feature spaces. The models include Copula-based RNNs using both HT and IPTW, PSM, logistic regression, and causal forests. Perturbation was introduced via Gaussian noise with a standard deviation equal to 10% of the feature scale to examine the robustness and sensitivity of model outputs to minor covariate variation. Table 1 includes the point estimate of ATE, the lower bound of the 95% confidence interval, and the upper bound of the 95% confidence interval.

Table 1. Comparison of Treatment Effect Estimates with Confidence Intervals (Original vs. Perturbed Models) for Simulated Data.

	Method	Estimate	LowerCI	UpperCI
1	Copula RNN with HT	0.7666	0.7734	0.8052
2	Copula RNN with HT (Perturbed)	0.8864	0.8736	0.8982
3	Copula RNN with IPTW	0.5945	0.5789	0.5943
4	Copula RNN with IPTW (Perturbed)	0.6561	0.6543	0.6581
5	PSM	1.0256	0.9628	1.0945
6	PSM (Perturbed)	1.0650	0.9884	1.1468
7	Logistic Regression	0.0704	0.0751	0.0888
8	Logistic Regression (Perturbed)	0.0820	0.0736	0.0897
9	Causal Forests	0.1611	0.1459	0.1758
10	Causal Forests (Perturbed)	0.1560	0.1414	0.1711

The sensitivity analysis reveals distinct behaviors across models under feature perturbation. For the Copula RNN with HT weights, the ATE increased from 0.7666 to 0.8864. The confidence intervals remained tight and non-overlapping. This behavior may be attributed to the model's flexibility and the inherent susceptibility of RNNs to overfitting.

In the Copula RNN with IPTW model, the ATE rose from 0.5945 to 0.6561 after perturbation. Although the confidence intervals were slightly shifted, they remained narrow, suggesting moderate sensitivity. This result reflects a trade-off between the flexibility of deep learning models and the stabilizing influence of inverse probability weighting.

The PSM method yielded the highest ATEs in both the original and perturbed settings (from 1.0256 to 1.0650). However, the associated confidence intervals were wide, reflecting higher variability and greater sensitivity to changes in covariates. This variability underscores the method's dependence on the quality of matching and the distribution of observed covariates.

Logistic regression produced conservative ATE estimates, with a slight increase from 0.0704 to 0.0820. The confidence intervals in both conditions were overlapping and narrow, suggesting strong robustness. However, this stability may come at the cost of underfitting, particularly when the underlying treatment-outcome relationships are nonlinear or involve complex interactions.

Causal forests demonstrated the most robust behavior, with ATEs of 0.1611 and 0.1560 under the original and perturbed settings, respectively. The confidence intervals overlapped almost completely, highlighting the model's insensitivity to minor feature perturbations. This robustness makes causal forests particularly well suited for noisy, real-world applications.

Overall, perturbation led to an increase in ATE estimates across most models—most notably in the Copula RNN approaches—while also reducing confidence interval widths, thereby implying greater precision. Logistic regression, although generally stable, showed signs of instability under some conditions, emphasizing the need for careful model selection in causal inference tasks.

Table 1 highlights the variation in ATE estimation across models. The Copula RNN with HT weights method yields the most precise and robust estimate, while PSM provides the largest estimate but with greater uncertainty. Logistic regression and causal forests produce smaller ATEs, possibly underestimating the true effect. Copula-based methods (HT and IPTW) consistently produce higher and more precise estimates, making them effective for treatment effect estimation. Researchers should carefully consider both the magnitude and precision of treatment effect estimates, especially in contexts where input features are subject to uncertainty or noise. The perturbation analysis reinforces the robustness of Copula-based methods and the resilience of causal forests, while highlighting potential sensitivity in traditional methods such as logistic regression.

4.2. Real Data Analysis

We downloaded a real dataset from https://github.com/propublica/compas-analysis/ (COMPAS dataset github site accessed on 19 March 2025), which is a sqlite3 database containing criminal history, jail and prison time, demographics, and COMPAS risk scores for defendants from Broward County.

The correlation matrix presented in Table 2 offers a comprehensive overview of the relationships among key demographic and criminal history variables, including age, sex, race, priors count, and decile score. Below is an interpretation of the key correlations:

The correlation between age and priors count is 0.1296, indicating a weak positive relationship. This suggests that individuals with more prior offenses tend to be slightly older. In contrast, the correlation between age and decile score is -0.3766, reflecting a moderate negative relationship. This implies that older individuals are generally assigned lower decile scores, potentially indicating a lower assessed risk of recidivism with increasing age.

Table 2. Correlation Matrix of Selected Variables.

	Age	Sex	Race	Priors_Count	Decile_Score
age	1.0000	0.0095	0.1291	0.1296	−0.3766
sex	0.0095	1.0000	−0.0172	0.1213	0.0508
race	0.1291	−0.0172	1.0000	−0.1954	−0.3094
priors_count	0.1296	0.1213	−0.1954	1.0000	0.4240
decile_score	−0.3766	0.0508	−0.3094	0.4240	1.0000

The correlation between sex and priors count is 0.1213, indicating a weak positive association, where males (coded as 1) are slightly more likely to have a higher number of prior offenses. The correlation between sex and decile score is 0.0508, which is near zero, suggesting no meaningful relationship between sex and assessed recidivism risk.

The correlation between race and priors count is −0.1954, denoting a weak negative relationship. This suggests that, on average, some racial groups may have fewer prior offenses. The correlation between race and decile score is −0.3094, indicating a moderate negative relationship. This implies that certain racial groups are more likely to receive lower recidivism risk scores, though the strength of this relationship remains modest.

A notable correlation exists between priors count and decile score (0.4240), suggesting a moderate positive association. This supports the expectation that individuals with more prior offenses are likely to be assessed as having a higher risk of recidivism.

Overall, the matrix reveals that age and priors count are moderately related to decile score, while race and sex show weaker associations. These results provide preliminary insights into the relationships among demographic and criminal history variables. However, to better understand the underlying mechanisms and control for potential confounders, further analysis—such as multivariate regression or causal modeling—is warranted.

This section provides a detailed explanation of the R code used for the analysis of treatment effects, focusing on various statistical and machine learning methods such as PSM, HT weights, Gaussian Copula transformation, and LSTM networks. The initial step involves cleaning and preparing the data for modeling. Relevant features, including `age`, `sex`, `race`, `priors_count`, and `decile_score`, are selected from the dataset. Any rows with missing values in these columns are removed using the `complete.cases()` function. Additionally, the treatment variable `is_recid` (indicating whether the individual recidivated or not) is converted into a factor for logistic regression modeling. This conversion is essential for subsequent analysis.

Propensity scores represent the likelihood of receiving a particular treatment (in this case, recidivism) based on observed covariates (such as `age`, `sex`, etc.). The code uses a logistic regression model to estimate these scores. The model predicts the probability of each individual receiving the treatment (`is_recid` = 1 for recidivism, 0 for non-recidivism) given the values of the covariates.

The model produces a set of probabilities (propensity scores) for each individual, which represent the likelihood of that individual receiving the treatment. These scores are crucial for the next step, where they are used to calculate the HT weights.

The HT weights are calculated using the inverse of the propensity scores. These weights adjust for selection bias or confounding by accounting for the treatment assignment probability. HT weights are computed by taking the reciprocal of the propensity score for each individual based on their treatment status (`is_recid`). For individuals who experienced recidivism (treatment group), the HT weight is $\frac{1}{\text{propensity score for recidivism}}$. For those who did not recidivate (control group), the HT weight is $\frac{1}{1-\text{propensity score for recidivism}}$. These weights ensure that individuals with a higher chance of receiving the treatment

(based on observed covariates) will have lower weights, while those less likely to receive the treatment will have higher weights, thus balancing the treatment groups. Gaussian Copula is used to model the dependencies between the covariates. A Copula is a statistical tool used to capture the relationships between multiple random variables. The Copula transforms the selected features into pseudo-observations. These pseudo-observations are uniformly distributed, allowing the model to focus on capturing dependencies between features rather than their original distributions. The Copula transformation is crucial because it helps model the complex interdependencies between the covariates, which is beneficial when training machine learning models like LSTM. The transformed features (pseudo-observations) are reshaped for input into the LSTM model. The LSTM model is trained to predict the decile score (which is related to recidivism risk) based on the transformed covariates. During training, HT weights are applied to the model to adjust for the different likelihoods of treatment assignment, ensuring that the model appropriately compensates for bias in treatment selection. The LSTM model consists of an LSTM layer with 64 units to process sequential data (even though in this case, the input is a single timestep per individual and dense layer with a linear activation function to output the final predicted value, which is the decile score for each individual.

After training the LSTM model, PSM is used to refine the treatment effect estimate. PSM is a statistical technique that matches individuals in the treatment group (those who recidivated) with those in the control group (those who did not) based on their propensity scores. This matching reduces bias by ensuring that individuals in both groups are similar on observed covariates. The matchit function is used to apply nearest neighbor matching. This technique pairs individuals with similar propensity scores. After matching, the ATE is calculated by comparing the average decile scores in the matched data.

To estimate heterogeneous treatment effects, we utilized the causal forest algorithm from the `grf` (Generalized Random Forests) framework. As a preprocessing step, all categorical variables in the dataset were first converted to factors and then numerically encoded. This transformation ensured compatibility with the causal forest model, which requires a numeric input for covariates. The covariate matrix was constructed from a subset of relevant features reflecting demographic and criminal history attributes. The outcome variable was defined as the decile risk score, which serves as a proxy for predicted recidivism risk. The binary treatment indicator was derived from the recidivism flag, where individuals flagged as recidivists were assigned a value of 1 and non-recidivists a value of 0. A causal forest model was then trained using 2000 trees, with a fixed random seed to ensure reproducibility of the results. After training, the model produced individualized treatment effect estimates for each observation. These estimates were averaged to obtain the sample-based average treatment effect (ATE), which quantifies the overall effect of the treatment (recidivism status) on the predicted risk score. This approach allows for flexible, nonparametric modeling of treatment effect heterogeneity and adjusts for complex interactions between covariates, offering a robust alternative to traditional parametric models in causal inference.

After obtaining treatment effect estimates from the all models given in this paper, we calculated confidence intervals for these estimates using bootstrapping, which is a resampling technique that helps estimate the variability of a statistic (in this case, the treatment effect estimate). By repeatedly sampling from the model predictions (with replacement), we computed the lower and upper percentiles (usually 95%) to form the confidence intervals. This process provides a range within which the true treatment effect is likely to lie, offering an understanding of the uncertainty associated with the estimate. Finally, the treatment effect estimates from the LSTM model (with HT weights,

PSM, causal forests, and logistic regression) are presented along with their respective confidence intervals. This allows for a comparison of the estimated treatment effects among the three methods. These techniques which we used enable a comprehensive treatment effect analysis, adjusting for confounding and selection biases, and providing confidence intervals for the estimated treatment effects. Figure 2 shows treatment effect estimates with bootstrapping 95% confidence intervals with COMPAS scores.

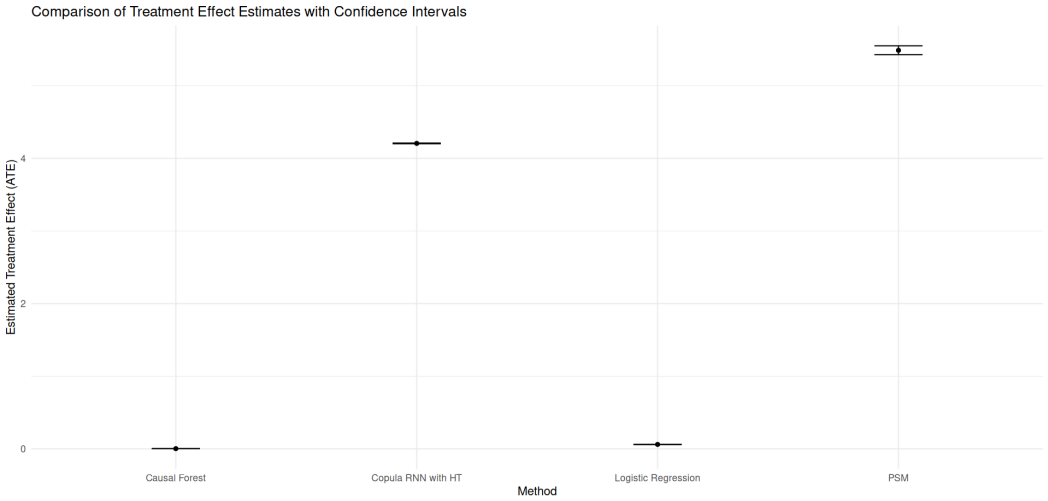


Figure 2. Treatment Effect Estimates with Confidence Intervals with COMPAS Scores.

Table 3 displays the average treatment effect (ATE) estimates with 95% confidence intervals, calculated using COMPAS scores. We evaluated and compared the treatment effect estimates using three statistical approaches: Copula RNN with HT weights, PSM, and logistic regression. These methods were employed to estimate the average treatment effect on recidivism risk, providing insights into each model's precision and reliability in this context.

Table 3. ATE Estimates with 95% Confidence Intervals from COMPAS Score Data.

Method	Estimate	LowerCI	UpperCI
Copula RNN with HT	4.2055	4.2009	4.2105
PSM	5.4868	5.4267	5.5488
Logistic Regression	0.0612	0.0595	0.0627
Causal Forests	0.0031	0.0029	0.0034

The Copula RNN with the HT weights method yielded an ATE of 4.2055, with a very narrow confidence interval of [4.2009, 4.2105]. This tight interval suggests high precision in estimation and indicates the model's robustness in capturing treatment effects. The substantial magnitude of the treatment effect, combined with the small uncertainty, supports the model's effectiveness in accounting for treatment assignment and adjusting for confounding in the COMPAS data.

The PSM approach reported an even higher treatment effect estimate of 5.4868, with a confidence interval of [5.4267, 5.5488]. While this method also provides a narrow interval, indicating reasonable precision, the wider range compared to the Copula RNN model may reflect sensitivity to the matching mechanism or remaining imbalance. Nonetheless, PSM's ability to reduce confounding through matching makes it a strong candidate for causal inference, especially in observational studies.

In contrast, the logistic regression model yielded a much smaller ATE of 0.0612, with a confidence interval of [0.0595, 0.0627]. Although the estimate is precise, as shown by

the narrow confidence interval, its magnitude is significantly lower than those produced by Copula RNN and PSM. This disparity may stem from logistic regression's reliance on strong parametric assumptions and its limited capacity to capture complex dependencies among covariates, especially when the treatment–outcome relationship is nonlinear or involves interactions.

Taken together, these results highlight that both the Copula RNN with HT weights and PSM models produce larger and statistically significant treatment effect estimates, implying stronger associations between the treatment (recidivism risk) and the outcome (decile score). The Copula RNN model's capacity to model complex covariate dependencies and the PSM model's strength in matching support their relative effectiveness in this context. On the other hand, logistic regression's more conservative estimate may indicate a more restricted view of the treatment effect under standard modeling assumptions.

All three models yield tight confidence intervals, demonstrating statistical significance and estimation stability. However, the marked differences in ATE magnitudes underscore the importance of model selection based on data complexity and the research objective. These findings suggest that more flexible models, such as Copula RNN and PSM, are advantageous when dealing with structured, nonlinear, or high-dimensional covariate interactions, which are common in criminal justice datasets like COMPAS. Future work may include sensitivity analyses and alternative modeling strategies to validate these insights and enhance treatment effect estimation in similar applied settings. The readers can reproduce these results by using R code in Appendix A.

5. Conclusions

In this study, we developed and evaluated advanced causal inference models incorporating Copula-based RNN with HT weights and IPTW and compared them against traditional methods such as PSM, logistic regression, and causal forests. Through extensive simulation studies, our Copula RNN model demonstrated superior ability to capture complex, nonlinear dependencies between covariates and treatment assignment, resulting in more accurate and precise estimation of average treatment effects (ATEs) under varying confounding scenarios. The simulation results highlighted the robustness of the proposed approach in settings with high-dimensional and dependent features, outperforming conventional parametric and semiparametric methods in bias reduction and confidence interval coverage.

Applying these methods to real-world COMPAS recidivism data further validated our findings. The Copula RNN with HT weights yielded substantial and statistically significant treatment effect estimates with tight confidence intervals, indicating high precision and robustness. Comparatively, PSM also produced large treatment effect estimates, while logistic regression estimates were markedly smaller, reflecting its limited flexibility in modeling complex treatment–covariate relationships inherent in real data. The variation in treatment effect magnitude across methods underscores the importance of flexible modeling frameworks capable of addressing nonlinearities and hidden dependencies for valid causal inference.

Overall, our integrated approach combining Copula theory with deep learning offers a powerful and versatile framework for treatment effect estimation in both simulated and observational study contexts. This work paves the way for future research into more sophisticated causal models leveraging machine learning, including extensions to time-varying treatments and multi-treatment regimes, as well as further methodological advancements to improve interpretability and computational efficiency. Practitioners applying causal inference to complex datasets, such as criminal justice risk assessments, may benefit from

adopting these advanced methodologies to achieve more reliable and insightful treatment effect estimates.

Funding: This research received no external funding.

Acknowledgments: We thank the five respected referees, Associated Editor and Editor for constructive and helpful suggestions which led to substantial improvement in the revised version. For the sake of transparency and reproducibility, the R code for this study can be found in the following GitHub repository: https://github.com/kjonomi/Rcode/blob/main/Axioms_causal (R code GitHub site) (accessed on 29 May 2025).

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. R Code

```
### Simulation with sensitivity analysis # Load Required Libraries
library(nnet)     # For multinomial logistic regression
library(keras)    # For deep learning models library(tensorflow)  #
For tensorflow backend library(MatchIt)   # For matching techniques
library(survival) library(dplyr) library(MASS)      # For correlated
data generation library(survminer) library(ggplot2)
library(survcomp)  # For C-index calculation library(kableExtra)
library(copula)    # For Gaussian copula transformation library(grf)
# For Causal Forest

# 1. Generate Highly Correlated Synthetic Data
set.seed(42)
n_samples <- 1000
n_features <- 6
base_corr <- 0.8
corr_matrix <-
matrix(base_corr, nrow = n_features, ncol = n_features)
diag(corr_matrix) <- 1
mean_values <- rep(0, n_features)

x_data <- mvrnorm(n = n_samples, mu = mean_values, Sigma =
corr_matrix)

x_data <- as.data.frame(x_data)

colnames(x_data) <- paste0(‘‘X’’, 1:n_features)

# 2. Add Nonlinear Transformations

x_data$X101 <- sin(x_data$X1) * cos(x_data$X2)

x_data$X102 <- log(abs(x_data$X3) + 1) * exp(-x_data$X4)

x_data$X103 <- (x_data$X5)^2 + (x_data$X6)^3

# 3. Generate Survival Data

shape_param <- 2 scale_param <- exp(0.5
```

```
* x_data$X1 + 0.3 * x_data$X2)

survival_times <- rweibull(n_samples, shape = shape_param, scale =
scale_param)

event_prob <- runif(n_samples)

event_type <- ifelse(event_prob < 0.5, 1, 2)

censoring_times <- runif(n_samples, min = 0, max =
max(survival_times) * 0.8)

observed_times <- pmin(survival_times, censoring_times)

observed_event <- ifelse(survival_times <= censoring_times,
event_type, 0)

synthetic_data <- cbind(x_data, time = observed_times, event =
observed_event)

synthetic_data <- as.data.frame(synthetic_data)

# 4. Multinomial Logistic Regression for Propensity Scores

synthetic_data$event <- as.factor(synthetic_data$event)
psm_model_multi <- nnet::multinom(event ~ ., data = synthetic_data)
prop_scores <- predict(psm_model_multi, type = ‘‘probs’’)

# 5. IPTW and HT Weights iptw_weights_1 <- 1 / prop_scores[, 1]
iptw_weights_2 <- 1 / prop_scores[, 2]

ht_weights <- 1 / (prop_scores[cbind(1:n_samples,
synthetic_data$event)])

sample_weights <- ifelse(synthetic_data$event == 1, iptw_weights_1,
iptw_weights_2)

# 6. Prepare Data

X_train <- as.matrix(synthetic_data[, 1:n_features])

y_train <- synthetic_data$time

# 7. Gaussian Copula Transformation

copula_model <- normalCopula(dim = n_features, dispstr = ‘‘un’’)
pseudo_obs <- pobs(X_train)

fit_result <- fitCopula(copula_model, pseudo_obs, method = ‘‘ml’’)
copula_transformed_features <- rCopula(n_samples, fit_result@copula)
X_train_rnn <- array(copula_transformed_features, dim = c(n_samples,
1, n_features))
```

```
# 8. Train LSTM with HT Weights
rnn_model_ht <- keras_model_sequential() %>%
layer_lstm(units = 64, return_sequences = FALSE,
input_shape = c(1, n_features)) %>%
layer_dense(units = 1, activation = ‘‘linear’’)
rnn_model_ht %>% compile(loss = "mean_squared_error",
optimizer = optimizer_adam(), metrics = c(‘‘mae’’))
rnn_model_ht %>% fit(X_train_rnn, y_train, epochs = 10,
batch_size = 64, sample_weight = ht_weights)

rnn_pred_ht <- predict(rnn_model_ht, X_train_rnn)

rnn_ate_ht <- mean(rnn_pred_ht)

# 9. Train LSTM with IPTW Weights
rnn_model_iptw <- keras_model_sequential() %>%
layer_lstm(units = 64, return_sequences = FALSE,
input_shape = c(1, n_features)) %>%
layer_dense(units = 1, activation = ‘‘linear’’)
rnn_model_iptw %>% compile(loss = "mean_squared_error",
optimizer = optimizer_adam(), metrics = c(‘‘mae’’))
rnn_model_iptw %>% fit(X_train_rnn, y_train, epochs = 10,
batch_size = 64, sample_weight = sample_weights)

rnn_pred_iptw <- predict(rnn_model_iptw, X_train_rnn)

rnn_ate_iptw <- mean(rnn_pred_iptw)

# 10. PSM

synthetic_data$event_binary <- ifelse(synthetic_data$event == 1, 1,
0)

psm_model <- matchit(event_binary ~ ., data = synthetic_data, method
= ‘‘nearest’’)

psm_matched_data <- match.data(psm_model)

psm_ate <- mean(psm_matched_data$time)

# 11. Causal Forest

numeric_data <- synthetic_data %>% mutate(across(where(is.factor),
~as.numeric(as.factor(.))))

X_cf <- as.matrix(numeric_data[, 1:n_features])

Y_cf <- synthetic_data$time

W_cf <- as.numeric(synthetic_data$event_binary)

cf_model <- causal_forest(X_cf, Y_cf, W = W_cf, num.trees = 2000,
seed = 42)
```

```
cf_pred <- predict(cf_model)$predictions cf_ate <- mean(cf_pred)


# 12. Bootstrap CI

bootstrap_ci <- function(x, n_bootstrap = 1000, conf_level = 0.95) {
boot_samples <- replicate(n_bootstrap, mean(sample(x, replace = TRUE)))
alpha <- (1 - conf_level) / 2
quantile(boot_samples, probs = c(alpha, 1 - alpha))
}

# --- Perturbation Function perturb_data <- function(data,
perturb_rate = 0.1) {
n_samples <- nrow(data)
n_features <- ncol(data)
perturbation <- rnorm(n_samples * n_features, mean = 0, sd = perturb_rate)
perturbed_data <- data + matrix(perturbation, nrow = n_samples,
ncol = n_features)
return(perturbed_data)
}

# --- RNN Models on Perturbed Data

X_train_perturbed <- perturb_data(X_train, perturb_rate = 0.1)
n_samples <- nrow(X_train_perturbed)

n_features <- ncol(X_train_perturbed)

X_train_rnn_perturbed<-array(perturb_data(copula_transformed_features,
perturb_rate = 0.1), dim = c(n_samples, 1, n_features))

rnn_model_ht %>% fit(X_train_rnn_perturbed, y_train,
epochs = 10, batch_size = 64, sample_weight = ht_weights)

rnn_pred_ht_perturbed <- predict(rnn_model_ht,
X_train_rnn_perturbed)

rnn_ate_ht_perturbed <- mean(rnn_pred_ht_perturbed)
rnn_ate_ht_perturbed_ci <- bootstrap_ci(rnn_pred_ht_perturbed)

rnn_model_iptw %>% fit(X_train_rnn_perturbed, y_train,
epochs = 10, batch_size = 64, sample_weight = sample_weights)
rnn_pred_iptw_perturbed <- predict(rnn_model_iptw,
X_train_rnn_perturbed)

rnn_ate_iptw_perturbed <- mean(rnn_pred_iptw_perturbed)
rnn_ate_iptw_perturbed_ci <- bootstrap_ci(rnn_pred_iptw_perturbed)

# --- PSM on Perturbed Data

synthetic_data_perturbed <- synthetic_data
synthetic_data_perturbed[, 1:n_features] <-
```

```
perturb_data(synthetic_data[, 1:n_features], perturb_rate = 0.1)
psm_model_perturbed <- matchit(event_binary ~ ., data =
synthetic_data_perturbed, method = ‘‘nearest’’)
psm_matched_data_perturbed <- match.data(psm_model_perturbed)
psm_ate_perturbed <- mean(psm_matched_data_perturbed$time)
psm_ate_perturbed_ci <-
bootstrap_ci(psm_matched_data_perturbed$time)

# --- Logistic Regression on Perturbed Data
psm_model_multi_perturbed <- nnet::multinom(event ~ ., data =
synthetic_data_perturbed)

logit_preds_perturbed <- predict(psm_model_multi_perturbed, type =
‘‘probs’’)

logit_ate_perturbed <- mean(logit_preds_perturbed[, 1])
logit_ate_perturbed_ci <- bootstrap_ci(logit_preds_perturbed[, 1])

# --- Causal Forest on Perturbed Data

X_cf_perturbed <- perturb_data(X_cf, perturb_rate = 0.1)
cf_model_perturbed <- causal_forest(X_cf_perturbed, Y = Y_cf, W =
W_cf, num.trees = 2000, seed = 42)

cf_pred_perturbed <- predict(cf_model_perturbed)$predictions
cf_ate_perturbed <- mean(cf_pred_perturbed)

cf_ate_perturbed_ci <- bootstrap_ci(cf_pred_perturbed)

# --- Results for Perturbed Models

ate_results_perturbed <-data.frame(Method = c(‘‘Copula RNN with HT (Perturbed)’’,
‘‘Copula RNN with IPTW (Perturbed)’’,
‘‘PSM (Perturbed)’’,
‘‘Logistic Regression (Perturbed)’’,
‘‘Causal Forest (Perturbed)’’),
Estimate = c(rnn_ate_ht_perturbed, rnn_ate_iptw_perturbed,
psm_ate_perturbed, logit_ate_perturbed, cf_ate_perturbed),
LowerCI = c(rnn_ate_ht_perturbed_ci[1], rnn_ate_iptw_perturbed_ci[1],
psm_ate_perturbed_ci[1],
logit_ate_perturbed_ci[1], cf_ate_perturbed_ci[1]),
UpperCI = c(rnn_ate_ht_perturbed_ci[2], rnn_ate_iptw_perturbed_ci[2],
psm_ate_perturbed_ci[2],
logit_ate_perturbed_ci[2], cf_ate_perturbed_ci[2])
)

# --- Visualization: Perturbed ATE Estimates
ggplot(ate_results_perturbed, aes(x = Method, y = Estimate)) +
geom_point() +
geom_errorbar(aes(ymin = LowerCI, ymax = UpperCI), width = 0.2) +
labs(title = ‘‘Comparison of Treatment Effect Estimates (Perturbed Models)’’,
y = ‘‘Estimated Treatment Effect (ATE)’’, x = ‘‘Method’’) +
theme_minimal() +
```

```
theme(axis.text.x = element_text(angle = 45, hjust = 1))

# --- Combined Table of Original + Perturbed Results
ate_results_combined <- data.frame(Method = c(‘‘Copula RNN with HT’’, ‘‘Copula RNN
with HT (Perturbed)’’,
‘‘Copula RNN with IPTW’’, ‘‘Copula RNN with IPTW (Perturbed)’’,
‘‘PSM’’, ‘‘PSM (Perturbed)’’,
‘‘Logistic Regression’’, ‘‘Logistic Regression (Perturbed)’’,
‘‘Causal Forest’’, ‘‘Causal Forest (Perturbed)’’),
Estimate = c(rnn_ate_ht, rnn_ate_ht_perturbed,
rnn_ate_iptw, rnn_ate_iptw_perturbed,
psm_ate, psm_ate_perturbed,
logit_ate, logit_ate_perturbed,
cf_ate, cf_ate_perturbed),
LowerCI = c(rnn_ate_ht_ci[1], rnn_ate_ht_perturbed_ci[1],
rnn_ate_iptw_ci[1], rnn_ate_iptw_perturbed_ci[1],
psm_ate_ci[1], psm_ate_perturbed_ci[1],
logit_ate_ci[1], logit_ate_perturbed_ci[1],
cf_ate_ci[1], cf_ate_perturbed_ci[1]),
UpperCI = c(rnn_ate_ht_ci[2], rnn_ate_ht_perturbed_ci[2],
rnn_ate_iptw_ci[2], rnn_ate_iptw_perturbed_ci[2],
psm_ate_ci[2], psm_ate_perturbed_ci[2],
logit_ate_ci[2], logit_ate_perturbed_ci[2],
cf_ate_ci[2], cf_ate_perturbed_ci[2])
)

# --- Display Combined Table
ate_results_combined %>%
kable(‘‘html’’, caption = ‘‘Comparison of Treatment Effect Estimates
(Original vs. Perturbed Models)’’) %>%
kable_styling(bootstrap_options = c(‘‘striped’’, ‘‘hover’’, ‘‘condensed’’))

print(ate_results_combined)


# Factor levels for better ordering

ate_results_combined$Method <- factor(ate_results_combined$Method,
levels = rev(ate_results_combined$Method))

# Highlight original vs. perturbed with color or linetype

ate_results_combined <- ate_results_combined %>%
mutate(ModelType = ifelse(grepl(‘‘Perturbed’’, Method), ‘‘Perturbed’’,
‘‘Original’’),
BaseMethod = gsub(‘‘ \\(Perturbed\\)’’, ‘‘’’, Method)
)

# Plot ggplot(ate_results_combined, aes(x = Estimate, y = Method,
color = ModelType)) +
geom_point(size = 3) +
geom_errorbarh(aes(xmin = LowerCI, xmax = UpperCI), height = 0.25) +
scale_color_manual(values = c(‘‘Original’’ = ‘‘#1f78b4’’,
```

```
‘‘Perturbed’’ = ‘‘#e31a1c’’)) +
labs(title = ‘‘Comparison of ATE Estimates: Original vs. Perturbed Models’’,
x = ‘‘Estimated Average Treatment Effect (ATE)’’,
y = ‘‘Method’’,
color = ‘‘Model Type’’) +
theme_minimal(base_size = 13) +
theme(legend.position = ‘‘top’’)



### Real Data
# Load required libraries

library(nnet)
library(keras)
library(tensorflow)
library(MatchIt)
library(survival)
library(dplyr)
library(MASS)
library(survminer)
library(ggplot2)
library(survcomp)
library(kableExtra)
library(copula)
library(boot)
library(grf)

# Load data
compas_data <- read.csv(‘‘compas-scores.csv’’)

# Select and clean features

features <- c(‘‘age’’, ‘‘sex’’, ‘‘race’’, ‘‘priors_count’’, ‘‘decile_score’’)
compas_data <- compas_data[complete.cases(compas_data[, features]),
]
compas_data$is_recid <- as.factor(compas_data$is_recid)

# Logistic regression for propensity

scores psm_model_multi <- nnet::multinom(is_recid ~ age + sex + race
+ priors_count + decile_score, data = compas_data)

prop_scores <- predict(psm_model_multi, type = ‘‘probs’’)

# IPTW and HT weights

iptw_weights_1 <- 1 / prop_scores[, 1]

iptw_weights_2 <- 1 / prop_scores[, 2]

ht_weights <- 1 / (prop_scores[cbind(1:nrow(compas_data),
as.numeric(compas_data$is_recid))])
```

```
sample_weights <- ifelse(compas_data$is_recid == 1, iptw_weights_1,
iptw_weights_2)

# Prepare training data

X_train <- as.matrix(compas_data[, features])

y_train <- compas_data$decile_score

# Copula transformation

copula_model <- normalCopula(dim = length(features), dispstr = ‘‘un’’)
pseudo_obs <- pobs(X_train)

fit_result <- fitCopula(copula_model, pseudo_obs, method = ‘‘ml’’)
copula_transformed_features <- rCopula(nrow(compas_data),
fit_result@copula)

X_train_rnn <- array(copula_transformed_features, dim =
c(nrow(compas_data), 1, length(features)))

# LSTM with HT weights

rnn_model_ht <- keras_model_sequential() %>%
layer_lstm(units = 64, return_sequences = FALSE,
input_shape = c(1, length(features))) %>%
layer_dense(units = 1, activation = ‘‘linear’’)
rnn_model_ht %>% compile(loss = "mean_squared_error",
optimizer = optimizer_adam(), metrics = c(‘‘mae’’))
rnn_model_ht %>% fit(X_train_rnn, y_train, epochs = 10,
batch_size = 64, sample_weight = ht_weights)

rnn_pred_ht <- predict(rnn_model_ht, X_train_rnn)

rnn_ate_ht <- mean(rnn_pred_ht)


# PSM
compas_data$event_binary <- ifelse(compas_data$is_recid == 1,
1, 0)

psm_model <- matchit(event_binary ~ age + sex + race + priors_count
+ decile_score, data = compas_data, method = ‘‘nearest’’)

psm_matched_data <- match.data(psm_model)

psm_ate <- mean(psm_matched_data$decile_score)


# Logistic regression

logit_preds <- predict(psm_model_multi, type = ‘‘probs’’)
```

```
# Causal Forest
numeric_data <- compas_data %>%
mutate(across(where(is.character), as.factor)) %>%
mutate(across(where(is.factor),
~as.numeric(as.factor(.))))  # One-hot equivalent

X_cf <- as.matrix(numeric_data[, features])

Y_cf <- compas_data$decile_score

W_cf <- as.numeric(compas_data$is_recid) - 1

cf_model <- causal_forest(X_cf, Y_cf, W = W_cf, num.trees = 2000,
seed = 42)

cf_pred <- predict(cf_model)$predictions

cf_ate <- mean(cf_pred)

# Bootstrap confidence interval

bootstrap_ci <- function(model_preds, n_bootstrap = 1000,
conf_level= 0.95) {
bootstrap_samples <- replicate(n_bootstrap,
sample(model_preds, replace = TRUE))
bootstrap_means <- apply(bootstrap_samples, 2, mean)
ci_lower <- quantile(bootstrap_means, (1 - conf_level) / 2)
ci_upper <- quantile(bootstrap_means, 1 - (1 - conf_level) / 2)
return(c(ci_lower, ci_upper))
}

# Confidence intervals

rnn_ate_ht_ci <- bootstrap_ci(rnn_pred_ht)

cf_ate_ci <- bootstrap_ci(cf_predictions$predictions)

psm_ate_ci <- bootstrap_ci(psm_matched_data$decile_score)

logit_ate_ci <- bootstrap_ci(logit_preds[, 1])


# Summary Table

ate_results <- data.frame(Method = c(‘‘Copula RNN with HT’’, ‘‘PSM’’,
‘‘Logistic Regression’’, ‘‘Causal Forest’’),
Estimate = c(rnn_ate_ht, psm_ate, mean(logit_prob), cf_ate),
LowerCI = c(rnn_ate_ht_ci[1], psm_ate_ci[1],
logit_ate_ci[1], cf_ate_ci[1]),
UpperCI = c(rnn_ate_ht_ci[2], psm_ate_ci[2],
logit_ate_ci[2], cf_ate_ci[2])
)
```

```
print(ate_results)

# 13. Visualization

ggplot(ate_results, aes(x = Method, y = Estimate)) +
geom_point() +
geom_errorbar(aes(ymin = LowerCI, ymax = UpperCI), width = 0.2) +
labs(title = ''Comparison of Treatment Effect Estimates
with Confidence Intervals'',
y = ''Estimated Treatment Effect (ATE)'',
x = ''Method'') +
theme_minimal()
```

References

1. Rubin, D.B. Estimating causal effects of treatments in randomized and nonrandomized studies. *J. Educ.* **1974**, *66*, 688–701. [CrossRef]
2. Rosenbaum, P.R.; Rubin, D.B. The central role of the propensity score in observational studies for causal effects. *Biometrika* **1983**, *70*, 41–55. [CrossRef]
3. Imbens, G.W. Nonparametric Estimation of Average Treatment Effects Under Exogeneity: A Review. *Rev. Econ. Stat.* **2004**, *86*, 4–29. [CrossRef]
4. Austin, P.C. An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivar. Behav. Res.* **2011**, *46*, 399–424. [CrossRef]
5. Anderson, C.; Rutkowski, L. *Multinomial Logistic Regression*; Osborne, J., Ed.; SAGE Publications, Inc.: Thousand Oaks, CA, USA, 2008; pp. 390–409.
6. Kim, J.-M. Integrating Copula-Based Random Forest and Deep Learning Approaches for Analyzing Heterogeneous Treatment Effects in Survival Analysis. *Mathematics* **2025**, *13*, 1659. [CrossRef]
7. Therneau, T.M.; Grambsch, P.M. *Modeling Survival Data: Extending the Cox Model*; Springer: New York, NY, USA, 2000.
8. Zhao, Q.; Hastie, T.; Tibshirani, R. Efficient computation of regularization paths for generalized additive models. *J. Comput. Graph. Stat.* **2019**, *28*, 727–744.
9. Wager, S.; Athey, S. Estimation and inference of heterogeneous treatment effects using random forests. *J. Am. Stat. Assoc.* **2018**, *113*, 1228–1242. [CrossRef]
10. Katzman, J.L.; Shaham, U.; Cloninger, A.; Bates, J.; Jiang, T.; Kluger, Y. DeepSurv: Personalized treatment recommender system using a Cox proportional hazards deep neural network. *BMC Med. Res. Methodol.* **2018**, *18*, 24. [CrossRef]
11. Hu, L.; Ji, J.; Li, F. Estimating heterogeneous survival treatment effect in observational data using machine learning. *Stat. Med.* **2021**, *40*, 4691–4713. [CrossRef]
12. Wang, Y.; Xie, J.; Zhao, X. DeepSurv landmarking: A deep learning approach for dynamic survival analysis with longitudinal data. *J. Stat. Comput. Simul.* **2024**, *95*, 186–207. [CrossRef]
13. Meyer, P.G.; Cherstvy, A.G.; Seckler, H.; Hering, R.; Blaum, N.; Jeltsch, F.; Metzler, R. Directedeness, correlations, and daily cycles in springbok motion: From data via stochastic models to movement prediction. *Phys. Rev. Res.* **2023**, *5*, 043129. [CrossRef]
14. Chen, Y.; Liu, Z.W.; Qin, Y. Finite-Time Topology Identification of Delayed Complex Dynamical Networks and Its Application. *Cyborg Bionic Syst.* **2024**, *5*, 92. [CrossRef]
15. Zhang, Y.; Wang, C.; Qi, J.; Peng, Y. Leave It to Large Language Models! Correction and Planning with Memory Integration. *Cyborg Bionic Syst.* **2024**, *5*, 87. [CrossRef] [PubMed]
16. Li, Q.; You, T.; Chen, J.; Zhang, Y.; Du, C. LI-EMRSQL: Linking Information Enhanced Text2SQL Parsing on Complex Electronic Medical Records. *IEEE Trans. Reliab.* **2024**, *73*, 1280–1290. [CrossRef]
17. Lou, Y.; Cheng, M.; Cao, Q.; Li, K.; Qin, H.; Bao, M.; Zhang, Y.; Lin, S.; Zhang, Y. Simultaneous quantification of mirabegron and vibegron in human plasma by HPLC-MS/MS and its application in the clinical determination in patients with tumors associated with overactive bladder. *J. Pharm. Biomed.* **2024**, *240*, 115937. [CrossRef]
18. Zhang, H.; Wang, L.-F.; Wang, X.-Q.; Deng, L.; He, B.-S.; Li, J.-M. Mechanisms and therapeutic potential of chinonin in nervous system diseases. *J. Asian Nat. Prod. Res.* **2024**, *26*, 1405–1420.

19. Fang, W.; Sun, W.; Fang, W.; Zhang, J.; Wang, C. Clinical features, treatment, and outcome of pembrolizumab induced cholangitis. Naunyn-Schmiedeberg's Arch. *Pharmacol* **2024**, *397*, 7905–7912.
20. Cheng, Z.; Wang, H.; Zhang, Y.; Ren, B.; Fu, Z.; Li, Z.; Tu, C. Deciphering the role of liquid-liquid phase separation in sarcoma: Implications for pathogenesis and treatment. *Cancer Lett.* **2025**, *616*, 217585. [CrossRef]
21. Joe, H. *Dependence Modeling with Copulas*; CRC Press: Boca Raton, FL, USA, 2014.
22. Nelsen, R.B. *An Introduction to Copulas*, 2nd ed.; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013.
23. Hernán, M.A.; Robins, J.M. *Causal Inference: What If*; Chapman & Hall/CRC: Boca Raton, FL, USA, 2020.
24. Athey, S.; Imbens, G. Recursive partitioning for heterogeneous causal effects. *Proc. Natl. Acad. Sci. USA* **2016**, *113*, 7353–7360. [CrossRef]
25. Sklar, A. Fonctions de répartition à n dimensions et leurs marges. *Publ. Inst. Stat. Univ.* **1959**, *8*, 229–231.
26. Efron, B. Better Bootstrap Confidence Intervals. *J. Am. Stat. Assoc.* **1987**, *82*, 171–185. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.