Package: MLCausal (via r-universe)

May 15, 2026

Type Package

Title Causal Inference Methods for Multilevel and Clustered Data

Version 0.1.0

Date 2026-04-07

Description Provides an end-to-end workflow for estimating average treatment effects in clustered (multilevel) observational data. Core functionality includes cluster-aware propensity score estimation using fixed effects and Mundlak-style specifications, inverse probability weighting, within-cluster nearest-neighbor matching, covariate balance diagnostics at both individual and cluster-mean levels, outcome regression with cluster-robust standard errors, propensity score overlap visualization, and tipping-point sensitivity analysis for omitted cluster-level confounding.

License MIT + file LICENSE

Encoding UTF-8

URL https://github.com/causalfragility-lab/MLCausal

BugReports https://github.com/causalfragility-lab/MLCausal/issues

Depends R (>= 4.1.0)

Imports stats, sandwich (>= 3.0-0), lmtest (>= 0.9-38), ggplot2 (>= 3.3.0), rlang (>= 0.4.0)

Suggests testthat (>= 3.0.0), knitr (>= 1.36), rmarkdown (>= 2.11)

VignetteBuilder knitr

RoxygenNote 7.3.3

Config/testthat/edition 3

Repository https://causalfragility-lab.r-universe.dev

Date/Publication 2026-04-08 20:16:35 UTC

RemoteUrl https://github.com/causalfragility-lab/mlcausal

RemoteRef HEAD

RemoteSha a0632ea1ed2525926f6a35d0512cc34f4c5577d5

Contents

`balance_ml` *Multilevel covariate balance diagnostics*

Description

Computes standardised mean differences (SMDs) at two levels simultaneously: the individual level and the cluster-mean level. Cluster-mean SMDs capture between-cluster imbalance that individual-level diagnostics miss in hierarchical data, and are the key diagnostic for assessing whether `ml_match`'s dual-balance optimisation has succeeded.

Usage

```
balance_ml(data, treatment, covariates, cluster, weights = NULL)
```

Arguments

`data`	A `data.frame`.
`treatment`	Character string. Name of the binary treatment variable (`0`/`1`).
`covariates`	Character vector of covariate names.
`cluster`	Character string. Name of the cluster identifier variable.
`weights`	Optional character string. Name of a weight variable in `data` (e.g. `"weights"` from `ml_weight` or `"match_weight"` from `ml_match`). `NULL` (default) computes unweighted SMDs.

Details

A common rule of thumb is $|\text{SMD}| < 0.10$ for adequate balance. Cluster-mean SMDs above this threshold indicate residual between-cluster confounding; in that case increase `lambda` in `ml_match` or switch to `method = "fixed"` in `ml_ps`.

When a cluster-mean SMD cannot be computed (e.g. because the matched sample contains only one treatment arm across clusters), the `smd` column returns the string `"Cluster-level SMD not estimable due to insufficient within-cluster variation after matching."` rather than `NA`. This makes the diagnostic failure explicit and actionable rather than silently missing.

Value

An object of class `balance_ml`, a named list:

overall Data frame with individual-level SMDs, one row per covariate. The `smd` column is numeric.

cluster_means Data frame with cluster-mean SMDs, one row per covariate. The `smd` column is character: a numeric value formatted to 4 decimal places, or a descriptive message when not estimable.

summary Row-bound combination of `overall` and `cluster_means`.

See Also

`ml_match`, `ml_weight`, `ml_ps`

Examples

```
dat   <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
ps    <- ml_ps(dat, "z", c("x1", "x2", "x3"), "school_id")
dat_w <- ml_weight(ps)
bal   <- balance_ml(dat_w, treatment = "z",
                    covariates = c("x1", "x2", "x3"),
                    cluster = "school_id", weights = "weights")
print(bal)
```

`estimate_att_ml` *Treatment effect estimation with cluster-robust standard errors*

Description

Fits a (weighted) linear outcome model and reports the treatment effect estimate with cluster-robust (HC1) standard errors via the sandwich estimator. When `covariates` and `weights` are both supplied the estimator is doubly robust.

Usage

```
estimate_att_ml(
  data,
  outcome,
  treatment,
  cluster,
  covariates = NULL,
  weights = NULL
)
```

Arguments

data	A data.frame. Typically the output of ml_weight or the data_matched element from ml_match.
outcome	Character string. Name of the continuous outcome variable.
treatment	Character string. Name of the binary treatment variable (0/1).
cluster	Character string. Name of the cluster identifier variable.
covariates	Optional character vector of regression covariates for doubly robust adjustment. NULL (default) fits treatment only.
weights	Optional character string. Name of a weight variable in data (e.g. "weights" or "match_weight"). NULL (default) fits an unweighted model.

Value

An object of class estimate_att_ml:

call The matched call.

model Fitted lm object.

vcov Cluster-robust variance-covariance matrix (HC1).

coef_table coeftest coefficient table.

estimate Point estimate for the treatment coefficient.

se Cluster-robust standard error.

p_value Two-sided p-value.

See Also

ml_weight, ml_match, sens_ml

Examples

```
dat   <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
ps    <- ml_ps(dat, "z", c("x1", "x2", "x3"), "school_id")
dat_w <- ml_weight(ps)
est   <- estimate_att_ml(dat_w, outcome = "y", treatment = "z",
                         cluster = "school_id", weights = "weights")
print(est)
```

`ml_match` *Within-cluster nearest-neighbour matching with dual-balance optimisation*

Description

Performs greedy nearest-neighbour matching within clusters using a composite distance that simultaneously targets balance on individual-level propensity scores *and* cluster-mean covariates. This dual-balance approach is the core methodological innovation of **MLCausal**: standard within-cluster PS matching achieves individual-level balance but often leaves substantial between-cluster (cluster-mean) imbalance. The composite distance penalises matches that worsen cluster-mean balance.

Usage

```
ml_match(ps_fit, ratio = 1L, replace = FALSE, caliper = NULL, lambda = 1)
```

Arguments

`ps_fit`	An object of class `ml_ps` from `ml_ps`.
`ratio`	Positive integer. Controls matched per treated unit. Default 1.
`replace`	Logical. Allow control reuse? Default `FALSE`.
`caliper`	Optional positive numeric. Maximum allowable logit-PS distance. `NULL` (default) applies no caliper.
`lambda`	Non-negative numeric. Weight on the cluster-mean balance penalty in the composite distance. `lambda = 0` gives standard PS matching; `lambda = 1` (default) gives equal weight to PS proximity and cluster-mean balance.

Value

An object of class `ml_match`, a named list:

call The matched call.

data_matched `data.frame` of matched units with columns `match_weight` (1 for treated, $1/k$ for controls) and `pair_id`.

pairs `data.frame` of matched pairs with `pair_id`, `treated_row`, `control_row`, `ps_distance`, and `composite_distance`.

treatment, cluster, ratio, replace, caliper, lambda Stored arguments.

n_unmatched Number of treated units that could not be matched.

Dual-balance distance

For each treated unit i and candidate control j in the same cluster g, the composite matching distance is

$$d_{ij} = |\text{logit}(\hat{e}_i) - \text{logit}(\hat{e}_j)| + \lambda \sum_k \frac{|\bar{x}_{k,g}^{(1)} - \bar{x}_{k,g}^{(0)}|}{s_k}$$

where $\hat{e}$ is the propensity score, $\bar{x}_{k,g}^{(t)}$ is the running cluster-g mean of covariate k for treatment arm t after each match is tentatively accepted, and s_k is the pooled SD of covariate k in the full sample. The tuning parameter λ (`lambda`) controls the weight given to cluster-mean balance relative to individual PS proximity. Setting `lambda = 0` recovers standard within-cluster PS matching.

See Also

`ml_ps`, `balance_ml`, `estimate_att_ml`

Examples

```
dat     <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
ps      <- ml_ps(dat, "z", c("x1", "x2", "x3"), "school_id")
matched <- ml_match(ps, ratio = 1, caliper = 0.5, lambda = 1)
print(matched)
```

`ml_ps` *Cluster-aware propensity score estimation*

Description

Estimates propensity scores for clustered observational data using one of three specifications to account for between-cluster confounding.

Usage

```
ml_ps(
  data,
  treatment,
  covariates,
  cluster,
  method = c("mundlak", "fixed", "single"),
  estimand = c("ATT", "ATE"),
  family = stats::binomial()
)
```

Arguments

`data`	A `data.frame` containing all required variables.
`treatment`	Character string. Name of the binary treatment variable (must be coded `0`/`1`).
`covariates`	Character vector of covariate names.
`cluster`	Character string. Name of the cluster identifier variable.

method	Character string. One of "mundlak" (default), "fixed", or "single".
estimand	Character string. Target estimand: "ATT" (default) or "ATE". Stored for downstream use by ml_weight and ml_match.
family	A GLM family object. Default stats::binomial().

Value

An object of class ml_ps, a named list with elements:

call The matched call.

model The fitted glm object.

data Working data frame with a .ps column of clipped propensity scores and, when method = "mundlak", _cm cluster-mean columns.

ps Numeric vector of clipped propensity scores.

treatment Name of the treatment variable.

covariates Names of the covariates used.

cluster Name of the cluster variable.

method PS estimation method used.

estimand Target estimand.

Methods

"mundlak" Augments the propensity score model with cluster means of each covariate (Mundlak terms). Softly absorbs between-cluster confounding without requiring within-cluster treatment variation in every cluster. **Recommended default.**

"fixed" Adds cluster indicators as a factor. Fully absorbs between-cluster confounding but requires within-cluster treatment variation in every cluster.

"single" Standard logistic regression with no cluster adjustment. Useful only as a naive baseline.

See Also

ml_weight, ml_match, plot_overlap_ml, balance_ml

Examples

```
dat <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
ps  <- ml_ps(dat, treatment = "z", covariates = c("x1", "x2", "x3"),
             cluster = "school_id", method = "mundlak", estimand = "ATT")
print(ps)
```

`ml_weight` *Multilevel inverse probability weights*

Description

Computes ATT or ATE inverse probability weights from propensity scores estimated by `ml_ps`. Stabilised weights (the default) are recommended to reduce variance without sacrificing consistency.

Usage

```
ml_weight(ps_fit, estimand = c("ATT", "ATE"), stabilize = TRUE, trim = NULL)
```

Arguments

`ps_fit`	An object of class `ml_ps` from `ml_ps`.
`estimand`	Character string. `"ATT"` (default) or `"ATE"`. Defaults to the estimand stored in `ps_fit`.
`stabilize`	Logical. `TRUE` (default) uses stabilised weights.
`trim`	Optional positive numeric. Weights above this value are Winsorised. `NULL` (default) applies no trimming.

Value

The working data frame from `ps_fit` with an appended `weights` column.

Weight formulae

Let $\hat{e}_i$ be the estimated propensity score and $\bar{p} = \Pr(Z = 1)$ the marginal treatment prevalence.

Unstabilised:

- ATT: $w = 1$ (treated), $w = \hat{e}/(1 - \hat{e})$ (control)
- ATE: $w = 1/\hat{e}$ (treated), $w = 1/(1 - \hat{e})$ (control)

Stabilised:

- ATT: $w = 1$ (treated), $w = \bar{p}\hat{e}/[(1 - \bar{p})(1 - \hat{e})]$ (control)
- ATE: $w = \bar{p}/\hat{e}$ (treated), $w = (1 - \bar{p})/(1 - \hat{e})$ (control)

See Also

`ml_ps`, `balance_ml`, `estimate_att_ml`

Examples

```
dat   <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
ps    <- ml_ps(dat, "z", c("x1", "x2", "x3"), "school_id")
dat_w <- ml_weight(ps, estimand = "ATT", stabilize = TRUE, trim = 10)
summary(dat_w$weights)
```

plot_overlap_ml *Plot propensity score overlap between treatment groups*

Description

Creates a density overlap plot of estimated propensity scores by treatment group. Substantial overlap supports the positivity assumption. The plot can be shown overall or faceted by cluster.

Usage

```
plot_overlap_ml(
  x,
  treatment = NULL,
  cluster = NULL,
  ps = NULL,
  facet_clusters = FALSE,
  top_n_clusters = 12L
)
```

Arguments

x	An ml_ps object from ml_ps, or a plain data.frame (in which case treatment, cluster, and ps must be supplied).
treatment	Character string. Treatment variable name. Required when x is a data.frame.
cluster	Character string. Cluster variable name. Required when x is a data.frame.
ps	Character string. Propensity score variable name. Required when x is a data.frame.
facet_clusters	Logical. Facet by cluster? Default FALSE.
top_n_clusters	Positive integer. Maximum clusters shown when facet_clusters = TRUE. Default 12.

Value

A ggplot object.

See Also

ml_ps

Examples

```
dat <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
ps  <- ml_ps(dat, "z", c("x1", "x2", "x3"), "school_id")
plot_overlap_ml(ps)
```

`sens_ml` *Tipping-point sensitivity analysis for omitted cluster confounding*

Description

Reports how strong a hypothetical omitted confounder (in units of the cluster-robust SE) would need to be to nullify the estimated treatment effect (push $|z_{\text{adj}}|$ below 1.96).

Usage

```
sens_ml(estimate, se, q = seq(0, 5, by = 0.1))
```

Arguments

`estimate`	Numeric scalar. Treatment effect estimate from `estimate_att_ml`.
`se`	Numeric scalar. Cluster-robust standard error of the estimate.
`q`	Numeric vector. Confounder strengths Γ to examine. Default `seq(0, 5, by = 0.1)`.

Value

A `data.frame` with columns:

confounder_strength The Γ value examined.

adjusted_estimate Estimate after subtracting assumed bias.

original_z Observed |estimate/SE|.

adjusted_z $\max(0,\, z_{\text{obs}} - \Gamma)$.

crosses_null `TRUE` when |adjusted_z| < 1.96.

Statistical model

The analysis assumes a linear bias model: an omitted variable U with confounder strength Γ shifts the estimate by $\delta = \Gamma \times \text{SE}$ and the z-statistic by Γ. This is transparent and easy to communicate, but is not a full formalisation. For rigorous sensitivity analysis see Cinelli & Hazlett (2020) or Rosenbaum bounds.

A message is printed if no tipping point is found within the examined range.

References

Cinelli, C. & Hazlett, C. (2020). Making sense of sensitivity: Extending omitted variable bias. *JRSS-B*, 82(1), 39–67. doi:10.1111/rssb.12348

See Also

`estimate_att_ml`

Examples

```
dat   <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
ps    <- ml_ps(dat, "z", c("x1", "x2", "x3"), "school_id")
dat_w <- ml_weight(ps)
est   <- estimate_att_ml(dat_w, "y", "z", "school_id",
                         weights = "weights")
sens  <- sens_ml(est$estimate, est$se)
head(sens)
```

`simulate_ml_data` *Simulate clustered observational data*

Description

Generates a multilevel dataset for prototyping and validating **MLCausal** workflows. The data-generating process (DGP) includes a cluster-level random effect that simultaneously drives treatment selection and the outcome (i.e. unmeasured cluster-level confounding when `method = "single"` is used in `ml_ps`), and heterogeneous treatment effects across clusters.

Usage

```
simulate_ml_data(
  n_clusters = 30L,
  cluster_size = 20L,
  n_min = 10L,
  seed = NULL
)
```

Arguments

`n_clusters`	Integer. Number of clusters (schools). Default `30`.
`cluster_size`	Integer. Expected cluster size drawn from $\text{Poisson}(\lambda = \texttt{cluster\_size} - 1) + 1$, then lower-bounded at `n_min`.
`n_min`	Integer. Minimum guaranteed cluster size. Prevents tiny clusters that lose entire treatment arms after matching. Default `10`.
`seed`	Optional integer. Random seed for reproducibility.

Value

A `data.frame` with columns:

school_id Integer cluster identifier (1 to `n_clusters`).

x1 Continuous individual-level covariate, $N(0, 1)$.

x2 Binary individual-level covariate, $\text{Bernoulli}(0.5)$.

x3 Continuous covariate correlated with the cluster random effect.

z Binary treatment indicator (1 = treated, 0 = control).

y Continuous outcome.

True estimands

Because treatment effect heterogeneity correlates with the cluster-level random effect, the true ATT $\neq$ ATE:

- **ATE** ≈ 0.5 (marginal mean of $\tau_j = 0.5 + 0.3u_j$).
- **ATT** > 0.5 because treated units are over-represented in high-u_j clusters where $\tau_j > 0.5$.

Examples

```
dat <- simulate_ml_data(n_clusters = 5, cluster_size = 10, seed = 1)
head(dat)
table(dat$z)
```

Index