Package: PartialTL (via r-universe)

May 23, 2026

Title Partial Transfer Learning for Causal Estimation

Version 0.1.0

Description Implements partial transfer learning (PTL) for causal effect estimation using source and target data, with bootstrap-based source detection. Provides data generating processes and nuisance functions for simulation.

License GPL (>= 2)

URL https://github.com/JackQuu/PartialTL

BugReports https://github.com/JackQuu/PartialTL/issues

Encoding UTF-8

Depends R (>= 4.0.0)

Imports MASS, DoubleML, ncvreg

Suggests lgr, mlr3, mlr3learners, glmnet

NeedsCompilation no

Author Xinhao Qu [aut, cre]

Maintainer Xinhao Qu <xqu018@ucr.edu>

Repository https://cran.r-universe.dev

Date/Publication 2026-02-18 19:10:13 UTC

RemoteUrl https://github.com/cran/PartialTL

RemoteRef HEAD

RemoteSha d17f8bf37de2b6e34900df896db852496e08b18b

Contents

`PartialTL-package` *Partial Transfer Learning for Causal Estimation and Source Detection*

Description

Implements partial transfer learning (PTL) and heterogeneous partial transfer learning (HPTL) for causal effect estimation using source and target data. Uses double machine learning for nuisance estimation and cross-fitting. Provides bootstrap-based source detection to identify which sources are transferable to the target. Includes data generating processes and nuisance functions for simulation.

Details

Main functions:

- `fit_PTL`: PTL causal estimate (single source).
- `fit_HPTL`: Heterogeneous PTL with multiple sources and covariate modules.
- `boot_detection`: Bootstrap-based source detection.
- `run_bootstrap_E_hat`: Bootstrap for nuisance function estimation.
- `DGP`: Data generating process for simulations.
- `RMSE`: Root mean squared error for evaluation.
- `f_0`, `g_0`, `f_k`, `g_k`: Nuisance functions for target and source models.

Run `demo(package = "PartialTL")` for demo scripts.

Author(s)

Author <author@example.com>

References

(Add references to the partial transfer learning and DML literature.)

boot_detection *Bootstrap-Based Source Detection*

Description

Identifies which sources are transferable to the target by comparing bootstrap estimates of the nuisance function on the target and each source. A source is detected as transferable if the difference is below a threshold proportional to the combined standard error.

Usage

```
boot_detection(D_t, X_t, Y_t, D_s_all, X_s_all, Y_s_all, source_sizes, B,
               ml_f, ml_g)
```

Arguments

D_t	Target treatment; $n_t \times 1$.
X_t	Target design matrix; $n_t \times p$.
Y_t	Target outcome; $n_t \times 1$.
D_s_all	Source treatments concatenated by row; rows split by source_sizes.
X_s_all	Source design matrices concatenated by row.
Y_s_all	Source outcomes concatenated by row.
source_sizes	Integer vector of length K: sample size of each source.
B	Number of bootstrap replications.
ml_f	Outcome learner for DoubleML.
ml_g	Treatment learner for DoubleML.

Value

A data frame with columns Source (label) and detected.source (list of logical vectors of length B per source).

See Also

run_bootstrap_E_hat, fit_PTL, fit_HPTL

DGP *Data Generating Process*

Description

Generates (X, D, Y) for simulations: multivariate normal X with AR(1)-type covariance, treatment D from a linear confounding model plus noise, and outcome Y = rho*D + f(X) + error.

Usage

```
DGP(n, q, p, p.nonzero, rho, Beta, Gamma, mu = rep(10, p), sigma = 0.5,
    f_func = f_0, g_func = g_0, seed = NULL)
```

Arguments

n	Sample size.
q	Causal dimension (kept for interface; unused in current body).
p	Nuisance dimension (number of covariates).
p.nonzero	Number of non-zero coefficients for Beta and Gamma.
rho	Causal effect (scalar).
Beta	Nuisance coefficient vector for outcome; $p \times 1$.
Gamma	Nuisance coefficient vector for treatment; $p \times 1$.
mu	Mean vector of X; length p.
sigma	Base for AR(1)-type covariance: $\Sigma_{i,j} = \sigma^{\lvert i-j \rvert}$.
f_func	Function f(X, Beta) for outcome nuisance; default f_0.
g_func	Function g(X, Gamma) for treatment equation; default g_0.
seed	Optional random seed.

Value

A list with components D (treatment), X (design matrix), Y (outcome), and data (cbind(Y, D, X)).

See Also

f_0, g_0, f_k, g_k, fit_PTL, RMSE

Examples

```
set.seed(1)
n <- 50
p <- 10
p.nz <- 3
Beta <- matrix(c(rep(0.5, p.nz), rep(0, p - p.nz)))
Gamma <- matrix(c(rep(0.3, p.nz), rep(0, p - p.nz)))
dat <- DGP(n, q = 1, p, p.nz, rho = 0.5, Beta, Gamma, seed = 1)
str(dat)
```

f_0 *Outcome Nuisance Function (Target Model)*

Description

Linear outcome nuisance function for the target model: $f_0(X, \beta) = X\beta$.

Usage

```
f_0(X, Beta)
```

Arguments

X	Design matrix; $n \times p$.
Beta	Coefficient vector (column matrix); $p \times 1$.

Value

Numeric vector of outcome nuisance values; length n.

See Also

g_0, f_k, g_k, DGP

f_k *Outcome Nuisance Function (Source Model)*

Description

Linear outcome nuisance function for source models: $f_k(X, \beta) = X\beta$. Same form as f_0; used in DGP for source data.

Usage

```
f_k(X, Beta)
```

Arguments

X	Design matrix; $n \times p$.
Beta	Coefficient vector (column matrix); $p \times 1$.

Value

Numeric vector of outcome nuisance values; length n.

See Also

f_0, g_k, DGP

`fit_HPTL` *HPTL (Heterogeneous Partial Transfer Learning) Fit*

Description

Fits heterogeneous partial transfer learning with multiple sources and covariate modules. Each source has its own module of covariates; PTL is applied per source and results are combined for the target causal estimate.

Usage

```
fit_HPTL(D_t, X_t, Y_t, D_s_all, X_s_all, Y_s_all, source_sizes, module_sizes,
         ml_f, ml_g, fold = 5)
```

Arguments

`D_t`	Target treatment; $n_t \times q$ matrix.
`X_t`	Target design matrix; $n_t \times p$ (same covariates as sources).
`Y_t`	Target outcome; $n_t \times 1$.
`D_s_all`	Source treatments concatenated by row; dimension is (sum of `source_sizes`) by q.
`X_s_all`	Source design matrices concatenated by row; rows split by `source_sizes`.
`Y_s_all`	Source outcomes concatenated by row; rows split by `source_sizes`.
`source_sizes`	Integer vector of length K: sample size of each source. Must sum to `nrow(Y_s_all)`.
`module_sizes`	Integer vector of length K: covariate module sizes. The k-th source uses columns `(cumsum(module_sizes)[k-1]+1):cumsum(module_sizes)[k]` of X.
`ml_f`	Outcome learner for DoubleML.
`ml_g`	Treatment learner for DoubleML.
`fold`	Number of folds for cross-fitting (default 5).

Value

A list with component `hat_rho_HPTL`: HPTL causal estimate on target.

See Also

`fit_PTL`, `boot_detection`

`fit_PTL` *PTL (Partial Transfer Learning) Fit*

Description

Fits partial transfer learning with cross-fitting and outcome reconstruction. Uses double machine learning on the source and SCAD on the outcome nuisance, then transfers to the target via the partial transfer formula.

Usage

```
fit_PTL(D_t, X_t, Y_t, D_s, X_s, Y_s, ml_f, ml_g, fold = 5L)
```

Arguments

`D_t`	Target treatment; $n_t \times 1$ matrix.
`X_t`	Target design matrix; $n_t \times p$.
`Y_t`	Target outcome; $n_t \times 1$.
`D_s`	Source treatment; $n_s \times 1$.
`X_s`	Source design matrix; $n_s \times p$.
`Y_s`	Source outcome; $n_s \times 1$.
`ml_f`	Outcome learner for DoubleML (e.g. `lrn("regr.cv_glmnet")`).
`ml_g`	Treatment learner for DoubleML (same type).
`fold`	Number of folds for cross-fitting (default 5).

Value

A list with components:

`hat_rho_s`	Source causal estimate.
`beta_hat_s`	Source nuisance coefficient estimate.
`E_s`	Estimated $E[Y - \rho D]$ on source (NA when `q != 1`).
`hat_rho_PTL`	PTL causal estimate on target.

See Also

`fit_HPTL`, `DGP`, `RMSE`

g_0 *Treatment Nuisance Function (Target Model)*

Description

Linear treatment/confounding function for the target model: $g_0(X, \gamma) = X\gamma$.

Usage

```
g_0(X, Gamma)
```

Arguments

X	Design matrix; $n \times p$.
Gamma	Coefficient vector (column matrix); $p \times 1$.

Value

Numeric vector of treatment equation values; length n.

See Also

f_0, f_k, g_k, DGP

g_k *Treatment Nuisance Function (Source Model)*

Description

Linear treatment/confounding function for source models: $g_k(X, \gamma) = X\gamma$. Same form as g_0; used in DGP for source data.

Usage

```
g_k(X, Gamma)
```

Arguments

X	Design matrix; $n \times p$.
Gamma	Coefficient vector (column matrix); $p \times 1$.

Value

Numeric vector of treatment equation values; length n.

See Also

g_0, f_k, DGP

RMSE *Root Mean Squared Error*

Description

Computes the root mean squared error of parameter estimates across simulations: for each component, $\sqrt{\text{mean}((\hat{\theta} - \theta)^2)}$.

Usage

```
RMSE(Theta, Theta.hat)
```

Arguments

Theta	True parameter vector; $p \times 1$.
Theta.hat	Matrix of estimates; $p \times n_{sim}$.

Value

Numeric vector of length p: RMSE per parameter component.

See Also

DGP, fit_PTL, fit_HPTL

Examples

```
th <- c(1, 2)
th_hat <- matrix(c(1.1, 2.2, 0.9, 1.8), nrow = 2)
RMSE(th, th_hat)
```

run_bootstrap_E_hat *Bootstrap for Nuisance Function Estimation*

Description

Runs bootstrap replications to estimate the nuisance quantity $E[Y - D\theta]$ using double machine learning, and returns the bootstrap distribution (mean and variance) of the estimate.

Usage

```
run_bootstrap_E_hat(D, X, Y, B, ml_f, ml_g)
```

Arguments

D	Treatment variable(s); matrix.
X	Design matrix.
Y	Outcome variable.
B	Number of bootstrap replications.
ml_f	Outcome learner for DoubleML.
ml_g	Treatment learner for DoubleML.

Value

A list with components:

E_hat	Numeric vector of length B: bootstrap estimates.
E_hat_mean	Mean of E_hat.
E_hat_var	Variance of E_hat.

See Also

boot_detection

Index