

Package ‘crossurr’

July 22, 2025

Type Package

Title Cross-Fitting for Doubly Robust Evaluation of High-Dimensional
Surrogate Markers

Version 1.1.2

Description Doubly robust methods for evaluating surrogate markers as outlined in: Agniel D, Hejblum BP, Thiebaut R & Parast L (2022).
``Doubly robust evaluation of high-dimensional surrogate markers", Biostatistics <doi:10.1093/biostatistics/kxac020>. You can use these methods to determine how much of the overall treatment effect is explained by a (possibly high-dimensional) set of surrogate markers.

License MIT + file LICENSE

Depends R (>= 3.6.0)

Imports dplyr, gbm, glmnet, glue, parallel, pbapply, purrr, ranger,
RCAL, rlang, SIS, stats, SuperLearner, tibble, tidyr

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Denis Agniel [aut, cre],
Boris P. Hejblum [aut],
Layla Parast [aut]

Maintainer Denis Agniel <dagniel@rand.org>

Repository CRAN

Date/Publication 2025-04-08 13:50:02 UTC

Contents

sim_data	2
xfr_surrogate	2
xf_surrogate	4
Index	7

sim_data	<i>A simple function to simulate example data.</i>
----------	--

Description

A simple function to simulate example data.

Usage

```
sim_data(n, p)
```

Arguments

n	number of simulated observations
p	number of simulated variables

Value

toy dataset used for demonstrating the methods with outcome y, treatment a, covariates x.1, x.2, and surrogates s.1, s.2, ...

xfr_surrogate	<i>A function for estimating the proportion of treatment effect explained using repeated cross-fitting.</i>
---------------	---

Description

A function for estimating the proportion of treatment effect explained using repeated cross-fitting.

Usage

```
xfr_surrogate(
  ds,
  x = NULL,
  s,
  y,
  a,
  splits = 50,
  K = 5,
  outcome_learners = NULL,
  ps_learners = NULL,
  interaction_model = TRUE,
  trim_at = 0.05,
  outcome_family = gaussian(),
  mthd = "superlearner",
```

```

    n_ptb = 0,
    ...
  )

```

Arguments

<code>ds</code>	a <code>data.frame</code> .
<code>x</code>	names of all covariates in <code>ds</code> that should be included to control for confounding (eg. age, sex, etc). Default is <code>NULL</code> .
<code>s</code>	names of surrogates in <code>ds</code> .
<code>y</code>	name of the outcome in <code>ds</code> .
<code>a</code>	treatment variable name (eg. groups). Expect a binary variable made of 1s and 0s.
<code>splits</code>	number of data splits to perform.
<code>K</code>	number of folds for cross-fitting. Default is 5.
<code>outcome_learners</code>	string vector indicating learners to be used for estimation of the outcome function (e.g., "SL.ridge"). See the SuperLearner package for details.
<code>ps_learners</code>	string vector indicating learners to be used for estimation of the propensity score function (e.g., "SL.ridge"). See the SuperLearner package for details.
<code>interaction_model</code>	logical indicating whether outcome functions for treated and control should be estimated separately. Default is <code>TRUE</code> .
<code>trim_at</code>	threshold at which to trim propensity scores. Default is <code>0.05</code> .
<code>outcome_family</code>	default is 'gaussian' for continuous outcomes. Other choice is 'binomial' for binary outcomes.
<code>mthd</code>	selected regression method. Default is 'superlearner', which uses the SuperLearner package for estimation. Other choices include 'lasso' (which uses <code>glmnet</code>), 'sis' (which uses SIS), 'cal' (which uses RCAL).
<code>n_ptb</code>	Number of perturbations. Default is 0 which means asymptotic standard errors are used.
<code>...</code>	additional parameters (in particular for <code>super_learner</code>)

Value

a tibble with columns:

- `Rm`: estimate of the proportion of treatment effect explained, computed as the median over the repeated splits.
- `R_se0` standard error for the PTE, accounting for the variability due to splitting.
- `R_cil0` lower confidence interval value for the PTE.
- `R_cih0` upper confidence interval value for the PTE.
- `Dm`: estimate of the overall treatment effect, computed as the median over the repeated splits.

- D_se0 standard error for the overall treatment effect, accounting for the variability due to splitting.
- D_cil0 lower confidence interval value for the overall treatment effect.
- D_cih0 upper confidence interval value for the overall treatment effect.
- Dsm: estimate of the residual treatment effect, computed as the median over the repeated splits.
- Ds_se0 standard error for the residual treatment effect, accounting for the variability due to splitting.
- Ds_cil0 lower confidence interval value for the residual treatment effect.
- Ds_cih0 upper confidence interval value for the residual treatment effect.

Examples

```
n <- 100
p <- 20
q <- 2
wds <- sim_data(n = n, p = p)

if(interactive()){
  lasso_est <- xfr_surrogate(ds = wds,
    x = paste('x.', 1:q, sep = ''),
    s = paste('s.', 1:p, sep = ''),
    a = 'a',
    y = 'y',
    splits = 2,
    K = 2,
    trim_at = 0.01,
    mthd = 'lasso',
    ncores = 1)
}
```

xf_surrogate

A function for estimating the proportion of treatment effect explained using cross-fitting.

Description

A function for estimating the proportion of treatment effect explained using cross-fitting.

Usage

```
xf_surrogate(
  ds,
  x = NULL,
  s,
  y,
  a,
```

```

K = 5,
outcome_learners = NULL,
ps_learners = outcome_learners,
interaction_model = TRUE,
trim_at = 0.05,
outcome_family = gaussian(),
mthd = "superlearner",
n_ptb = 0,
ncores = parallel::detectCores() - 1,
...
)

```

Arguments

<code>ds</code>	a <code>data.frame</code> .
<code>x</code>	names of all covariates in <code>ds</code> that should be included to control for confounding (eg. age, sex, etc). Default is <code>NULL</code> .
<code>s</code>	names of surrogates in <code>ds</code> .
<code>y</code>	name of the outcome in <code>ds</code> .
<code>a</code>	treatment variable name (eg. groups). Expect a binary variable made of 1s and 0s.
<code>K</code>	number of folds for cross-fitting. Default is 5.
<code>outcome_learners</code>	string vector indicating learners to be used for estimation of the outcome function (e.g., "SL.ridge"). See the SuperLearner package for details.
<code>ps_learners</code>	string vector indicating learners to be used for estimation of the propensity score function (e.g., "SL.ridge"). See the SuperLearner package for details.
<code>interaction_model</code>	logical indicating whether outcome functions for treated and control should be estimated separately. Default is <code>TRUE</code> .
<code>trim_at</code>	threshold at which to trim propensity scores. Default is 0.05.
<code>outcome_family</code>	default is 'gaussian' for continuous outcomes. Other choice is 'binomial' for binary outcomes.
<code>mthd</code>	selected regression method. Default is 'superlearner', which uses the SuperLearner package for estimation. Other choices include 'lasso' (which uses <code>glmnet</code>), 'sis' (which uses <code>SIS</code>), 'cal' (which uses <code>RCAL</code>).
<code>n_ptb</code>	Number of perturbations. Default is 0 which means asymptotic standard errors are used.
<code>ncores</code>	number of CPUs used for parallel computations. Default is <code>parallel::detectCores()-1</code>
<code>...</code>	additional parameters (in particular for <code>super_learner</code>)

Value

a tibble with columns:

- R: estimate of the proportion of treatment effect explained, equal to $1 - \text{deltahat}_s / \text{deltahat}$.
- R_se standard error for the PTE.
- deltahat_s: residual treatment effect estimate.
- deltahat_s_se: standard error for the residual treatment effect.
- pi_o: estimate of the proportion of overlap.
- R_o: PTE only in the overlap region.
- R_o_se: the standard error for R_o.
- deltahat_s_o: residual treatment effect in overlap region,
- deltahat_s_se_o: standard error for deltahat_s_o.
- deltahat: overall treatment effect estimate.
- deltahat_se: standard error for overall treatment effect estimate.
- delta_diff: difference between the treatment effects, equal to the numerator of PTE.
- dd_se: standard error for delta_diff

Examples

```
n <- 300
p <- 50
q <- 2
wds <- sim_data(n = n, p = p)

if(interactive()){
  sl_est <- xf_surrogate(ds = wds,
    x = paste('x.', 1:q, sep = '.'),
    s = paste('s.', 1:p, sep = '.'),
    a = 'a',
    y = 'y',
    K = 4,
    trim_at = 0.01,
    mthd = 'superlearner',
    outcome_learners = c("SL.mean", "SL.lm", "SL.svm", "SL.ridge"),
    ps_learners = c("SL.mean", "SL.glm", "SL.svm", "SL.lda"),
    ncores = 1)

  lasso_est <- xf_surrogate(ds = wds,
    x = paste('x.', 1:q, sep = '.'),
    s = paste('s.', 1:p, sep = '.'),
    a = 'a',
    y = 'y',
    K = 4,
    trim_at = 0.01,
    mthd = 'lasso',
    ncores = 1)
}
```

Index

`sim_data`, [2](#)

`xf_surrogate`, [4](#)

`xfr_surrogate`, [2](#)