Package: precmed (via r-universe)

March 5, 2025

Type Package

Title Precision Medicine

Version 1.1.0.9000

Description A doubly robust precision medicine approach to fit, cross-validate and visualize prediction models for the conditional average treatment effect (CATE). It implements doubly robust estimation and semiparametric modeling approach of treatment-covariate interactions as proposed by Yadlowsky et al. (2020) <doi:10.1080/01621459.2020.1772080>.

Depends R (>= 3.5.0)

License Apache License (== 2.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports dplyr, gbm, gam, ggplot2, glmnet, graphics, MASS, mgcv, rlang, stringr, tidyr, survival, randomForestSRC

NeedsCompilation no

BugReports https://github.com/smartdata-analysis-and-statistics/precmed/issues

URL https://github.com/smartdata-analysis-and-statistics/precmed, https://smartdata-analysis-and-statistics.github.io/precmed/

Config/pak/sysreqs libglpk-dev make libicu-dev libxml2-dev libx11-dev

Repository https://smartdata-analysis-and-statistics.r-universe.dev

RemoteUrl https://github.com/smartdata-analysis-and-statistics/precmed

RemoteRef HEAD

RemoteSha 3203115fbfcc3bc1051895d9e1b927c3f61d2bf0

Contents

---

abc *Compute the area between curves from the* "precmed" *object*

Description

Compute the area between curves (ABC) for each scoring method in the "precmed" object. This should be run only after results of catecv() have been obtained.

Usage

```
abc(x, ...)

## Default S3 method:
abc(x, ...)
```

Arguments

x	An object of class "precmed".
...	Additional arguments (currently unused).

Details

The ABC is the area between a validation curve and the overall ATE in the validation set. It is calculated for each scoring method separately. Higher ABC values are preferable as they indicate that more treatment effect heterogeneity is captured by the scoring method. Negative values of ABC are possible if segments of the validation curve cross the overall ATE line. The ABC is calculated with the auc() in utility.R with a natural cubic spline interpolation. The calculation of the ABC is always based on validation curves based on 100 proportions equally spaced from min(prop.cutoff) to max(prop.cutoff).

The ABC is a metric to help users select the best scoring method in terms of capturing treatment effect heterogeneity in the data. It should be used in complement to the visual inspection of the validation curves in the validation set in plot().

Value

Returns a matrix of numeric values with number of columns equal to the number cross-validation iteration and number of rows equal to the number of scoring methods in `x`.

References

Zhao, L., Tian, L., Cai, T., Claggett, B., & Wei, L. J. (2013). *Effectively selecting a target population for a future comparative study. Journal of the American Statistical Association, 108(502), 527-539.*

See Also

`catecv()` function and `plot()`, `boxplot()` methods for `"precmed"` objects.

Examples

```
# Count outcome
cv_count <- catecv(response = "count",
                   data = countExample,
                   score.method = "poisson",
                   cate.model = y ~ age + female + previous_treatment +
                                previous_cost + previous_number_relapses +
                                offset(log(years)),
                   ps.model = trt ~ age + previous_treatment,
                   higher.y = FALSE, cv.n = 5, verbose = 1)

# ABC of the validation curves for each method and each CV iteration
abc(cv_count)

# Survival outcome
library(survival)
cv_surv <- catecv(response = "survival",
                  data = survivalExample,
                  score.method = c("poisson", "randomForest"),
                  cate.model = Surv(y, d) ~ age + female + previous_cost +
                               previous_number_relapses,
                  ps.model = trt ~ age + previous_treatment,
                  higher.y = FALSE,
                  cv.n = 5)

# ABC of the validation curves for each method and each CV iteration
abc(cv_surv)
```

`abc.precmed` *Compute the area between curves from the* `"precmed"` *object*

Description

Compute the area between curves (ABC) for each scoring method in the "precmed" object. This should be run only after results of catecv() have been obtained.

Usage

```
## S3 method for class 'precmed'
abc(x, ...)
```

Arguments

x	An object of class "precmed".
...	Additional arguments (currently unused).

Details

The ABC is the area between a validation curve and the overall ATE in the validation set. It is calculated for each scoring method separately. Higher ABC values are preferable as they indicate that more treatment effect heterogeneity is captured by the scoring method. Negative values of ABC are possible if segments of the validation curve cross the overall ATE line. The ABC is calculated with the auc() in utility.R with a natural cubic spline interpolation. The calculation of the ABC is always based on validation curves based on 100 proportions equally spaced from min(prop.cutoff) to max(prop.cutoff).

The ABC is a metric to help users select the best scoring method in terms of capturing treatment effect heterogeneity in the data. It should be used in complement to the visual inspection of the validation curves in the validation set in plot().

Value

Returns a matrix of numeric values with number of columns equal to the number cross-validation iteration and number of rows equal to the number of scoring methods in x.

References

Zhao, L., Tian, L., Cai, T., Claggett, B., & Wei, L. J. (2013). *Effectively selecting a target population for a future comparative study. Journal of the American Statistical Association, 108(502), 527-539.*

See Also

catecv() function and plot(), boxplot() methods for "precmed" objects.

Examples

```
# Count outcome
cv_count <- catecv(response = "count",
                   data = countExample,
                   score.method = "poisson",
                   cate.model = y ~ age + female + previous_treatment +
                                previous_cost + previous_number_relapses +
```

```
                          offset(log(years)),
                 ps.model = trt ~ age + previous_treatment,
                 higher.y = FALSE, cv.n = 5, verbose = 1)

# ABC of the validation curves for each method and each CV iteration
abc(cv_count)

# Survival outcome
library(survival)
cv_surv <- catecv(response = "survival",
                  data = survivalExample,
                  score.method = c("poisson", "randomForest"),
                  cate.model = Surv(y, d) ~ age + female + previous_cost +
                               previous_number_relapses,
                  ps.model = trt ~ age + previous_treatment,
                  higher.y = FALSE,
                  cv.n = 5)

# ABC of the validation curves for each method and each CV iteration
abc(cv_surv)
```

`arg.checks` *Check arguments Catered to all types of outcome Apply at the beginning of* `pmcount()`*,* `cvcount()`*,* `drcount.inference()`*,* `catefitsurv()`*,* `catecvsurv()`*, and* `drsurv.inference()`

Description

Check arguments Catered to all types of outcome Apply at the beginning of `pmcount()`, `cvcount()`, `drcount.inference()`, `catefitsurv()`, `catecvsurv()`, and `drsurv.inference()`

Usage

```
arg.checks(
  fun,
  response,
  data,
  followup.time = NULL,
  tau0 = NULL,
  surv.min = NULL,
  ipcw.method = NULL,
  ps.method,
  minPS,
  maxPS,
  higher.y = NULL,
  score.method = NULL,
```

```
  abc = NULL,
  prop.cutoff = NULL,
  prop.multi = NULL,
  train.prop = NULL,
  cv.n = NULL,
  error.max = NULL,
  max.iter = NULL,
  initial.predictor.method = NULL,
  tree.depth = NULL,
  n.trees.rf = NULL,
  n.trees.boosting = NULL,
  B = NULL,
  Kfold = NULL,
  plot.gbmperf = NULL,
  error.maxNR = NULL,
  max.iterNR = NULL,
  tune = NULL,
  n.boot = NULL,
  plot.boot = NULL,
  interactions = NULL
)
```

Arguments

fun	A function for which argument check is needed; "catefit" for `catefitcount()` and `catefitsurv()`, "crossv" for `catecvcount()` and `catecvsurv()`, and "drinf" for `drcount.inference()` and `drsurv.inference()`. No default.
response	The type of response. Always 'survival' for this function.
data	A data frame containing the variables in the outcome and propensity score models; a data frame with `n` rows (1 row per observation).
followup.time	Follow-up time, interpreted as the potential censoring time. If the potential censoring time is known, followup.time is the name of a corresponding column in the data. Otherwise, set `followup.time == NULL`.
tau0	The truncation time for defining restricted mean time lost.
surv.min	Lower truncation limit for probability of being censored (positive and very close to 0).
ipcw.method	The censoring model. Allowed values are: `'breslow'` (Cox regression with Breslow estimator of the baseline survivor function), `'aft (exponential)'`, `'aft (weibull)'`, `'aft (lognormal)'` or `'aft (loglogistic)'`. Default is `'breslow'`.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: `'glm'` for logistic regression with main effects only (default), or `'lasso'` for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in `ps.model`). Relevant only when `ps.model` has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is `0.01`.

maxPS — A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

higher.y — A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.

score.method — A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'boosting', 'poisson', 'twoReg', 'contrastReg', 'negBin'. Default specifies all 5 methods.

abc — A logical value indicating whether the area between curves (ABC) should be calculated at each cross-validation iterations, for each score.method. Default is TRUE.

prop.cutoff — A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cut-off to separate observations in nested subgroups (below vs above cutoff). The length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).

prop.multi — A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of length(prop.multi) > 2. Each element represents the cutoff to separate the observations into length(prop.multi) - 1 mutually exclusive subgroups. Default is c(0, 1/3, 2/3, 1).

train.prop — A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is 3/4.

cv.n — A positive integer value indicating the number of cross-validation iterations. Default is 10.

error.max — A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is 0.1.

max.iter — A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is 5,000.

initial.predictor.method — A character vector for the method used to get initial outcome predictions conditional on the covariates in cate.model in score.method = 'twoReg' and 'contrastReg'. Allowed values include one of 'randomForest', 'boosting' and 'logistic' (fastest). Default is 'randomForest'.

tree.depth — A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 2.

n.trees.rf — A positive integer specifying the maximum number of trees in random forest. Used if score.method = 'ranfomForest' or if initial.predictor.method = 'randomForest' with score.method = 'twoReg' or 'contrastReg'. Default is 1000.

`n.trees.boosting`
: A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if `score.method = 'boosting'` or if `initial.predictor.method = 'boosting'` with `score.method = 'twoReg'` or `'contrastReg'`. Default is `150`.

`B`
: A positive integer specifying the number of time cross-fitting is repeated in `score.method = 'twoReg'` and `'contrastReg'`. Default is `3`.

`Kfold`
: A positive integer specifying the number of folds (parts) used in cross-fitting to partition the data in `score.method = 'twoReg'` and `'contrastReg'`. Default is `6`.

`plot.gbmperf`
: A logical value indicating whether to plot the performance measures in boosting. Used only if `score.method = 'boosting'` or if `score.method = 'twoReg'` or `'contrastReg'` and `initial.predictor.method = 'boosting'`. Default is `TRUE`.

`error.maxNR`
: A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if `score.method = 'contrastReg'`. Default is `0.001`.

`max.iterNR`
: A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if `score.method = 'contrastReg'`. Default is `150`.

`tune`
: A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. `tune[1]` is the step size, `tune[2]` specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if `score.method = 'contrastReg'`. Default is `c(0.5, 2)`.

`n.boot`
: A numeric value indicating the number of bootstrap samples used. This is only relevant if `inference = TRUE`. Default is `500`.

`plot.boot`
: A logic value indicating whether histograms of the bootstrapped log(rate ratio) should be produced at every `n.boot/10`-th iteration and whether the final histogram should be outputted. Default is `FALSE`.

`interactions`
: A logical value indicating whether the outcome model should assume interactions between `x` and `trt`. If `TRUE`, interactions will be assumed only if at least 10 patients received each treatment option. Default is `TRUE`.

Value

Nothing. Will stop if arguments are incorrect.

`arg.checks.common` *Check arguments that are common to all types of outcome USed inside* `arg.checks()`

Description

Check arguments that are common to all types of outcome USed inside `arg.checks()`

Usage

```
arg.checks.common(
  fun,
  ps.method,
  minPS,
  maxPS,
  higher.y = NULL,
  abc = NULL,
  prop.cutoff = NULL,
  prop.multi = NULL,
  B = NULL,
  Kfold = NULL,
  plot.gbmperf = NULL,
  tree.depth = NULL,
  n.trees.boosting = NULL,
  error.maxNR = NULL,
  max.iterNR = NULL,
  tune = NULL,
  train.prop = NULL,
  cv.n = NULL,
  error.max = NULL,
  max.iter = NULL,
  n.boot = NULL,
  plot.boot = NULL
)
```

Arguments

fun	A function for which argument check is needed; "pm" for pmcount(), "cv" for cvcount(), and "drinf" for drcount.inference(). No default.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
higher.y	A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.
abc	A logical value indicating whether the area between curves (ABC) should be calculated at each cross-validation iterations, for each score.method. Default is TRUE.
prop.cutoff	A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cut-off to separate observations in nested subgroups (below vs above cutoff). The

length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).

prop.multi A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of length(prop.multi) > 2. Each element represents the cutoff to separate the observations into length(prop.multi) - 1 mutually exclusive subgroups. Default is c(0, 1/3, 2/3, 1).

B A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold A positive integer specifying the number of folds (parts) used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 6.

plot.gbmperf A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

tree.depth A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 2.

n.trees.boosting
A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 200.

error.maxNR A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

max.iterNR A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.

tune A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

train.prop A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is 3/4.

cv.n A positive integer value indicating the number of cross-validation iterations. Default is 10.

error.max A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is 0.1.

max.iter A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is 5,000.

n.boot	A numeric value indicating the number of bootstrap samples used. This is only relevant if inference = TRUE. Default is 500.
plot.boot	A logic value indicating whether histograms of the bootstrapped log(rate ratio) should be produced at every n.boot/10-th iteration and whether the final histogram should be outputted. Default is FALSE.

Value

Nothing. Will stop if arguments are incorrect.

---

atefit *Doubly robust estimator of and inference for the average treatment effect for count, survival and continuous data*

Description

Doubly robust estimator of the average treatment effect between two treatments, which is the rate ratio for count outcomes, the restricted mean time lost ratio for survival outcomes and the mean difference for continuous outcome. Bootstrap is used for inference.

Usage

```
atefit(
  response,
  data,
  cate.model,
  ps.model,
  ps.method = "glm",
  ipcw.model = NULL,
  ipcw.method = "breslow",
  minPS = 0.01,
  maxPS = 0.99,
  followup.time = NULL,
  tau0 = NULL,
  surv.min = 0.025,
  interactions = TRUE,
  n.boot = 500,
  seed = NULL,
  verbose = 0
)
```

Arguments

response	A string describing the type of outcome in the data. Allowed values include "count" (see catecvcount()), "survival" (see catecvsurv()) and "continuous" (see catecvmean()).

data	A data frame containing the variables in the outcome, propensity score, and inverse probability of censoring models (if specified); a data frame with n rows (1 row per observation).
cate.model	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side. For survival outcomes, a Surv object must be used to describe the outcome.
ps.model	A formula describing the propensity score (PS) model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a randomized controlled trial, specify ps.model = ~1 as an intercept-only model.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
ipcw.model	A formula describing the inverse probability of censoring weighting (IPCW) model to be fitted. The left-hand side must be empty. Only applies for survival outcomes. Default is NULL, which corresponds to specifying the IPCW with the same covariates as the outcome model cate.model, plus the treatment.
ipcw.method	A character value for the censoring model. Only applies for survival outcomes. Allowed values are: 'breslow' (Cox regression with Breslow estimator of t he baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)' (accelerated failure time model with different distributions for y variable). Default is 'breslow'.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
followup.time	A column name in data specifying the maximum follow-up time, interpreted as the potential censoring time. Only applies for survival outcomes. Default is NULL, which corresponds to unknown potential censoring time.
tau0	The truncation time for defining restricted mean time lost. Only applies for survival outcomes. Default is NULL, which corresponds to setting the truncation time as the maximum survival time in the data.
surv.min	Lower truncation limit for the probability of being censored. It must be a positive value and should be chosen close to 0. Only applies for survival outcomes. Default is 0.025.
interactions	A logical value indicating whether the outcome model should assume interactions between x and trt. Applies only to count outcomes. If TRUE, interactions will be assumed only if at least 10 patients received each treatment option. Default is TRUE.
n.boot	A numeric value indicating the number of bootstrap samples used. Default is 500.
seed	An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.

verbose — An integer value indicating whether intermediate progress messages and histograms should be printed. 1 indicates messages are printed and 0 otherwise. Default is 0.

Details

For count response, see details in atefitcount(). For survival response, see details in atefitsurv().

Value

For count response, see description of outputs in atefitcount(). For survival response, see description of outputs in atefitsurv().

Examples

```
# Count outcome
output <- atefit(response = "count",
                 data = countExample,
                 cate.model = y ~ age + female + previous_treatment +
                              previous_cost + previous_number_relapses +
                              offset(log(years)),
                 ps.model = trt ~ age + previous_treatment,
                 n.boot = 50,
                 seed = 999)
output
plot(output)



# Survival outcome
tau0 <- with(survivalExample,
               min(quantile(y[trt == "drug1"], 0.95), quantile(y[trt == "drug0"], 0.95)))

output2 <- atefit(response = "survival",
                  data = survivalExample,
                  cate.model = survival::Surv(y, d) ~ age + female +
                        previous_cost + previous_number_relapses,
                        ps.model = trt ~ age + previous_treatment,
                  tau0 = tau0,
                  seed = 999)
output2
plot(output2)
```

---

atefitcount — *Doubly robust estimator of and inference for the average treatment effect for count data*

Description

Doubly robust estimator of the average treatment effect between two treatments, which is the rate ratio for count outcomes. Bootstrap is used for inference.

Usage

```
atefitcount(
  data,
  cate.model,
  ps.model,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  interactions = TRUE,
  n.boot = 500,
  seed = NULL,
  verbose = 0
)
```

Arguments

`data`	A data frame containing the variables in the outcome, propensity score, and inverse probability of censoring models (if specified); a data frame with `n` rows (1 row per observation).
`cate.model`	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
`ps.model`	A formula describing the propensity score (PS) model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0 or 1. If data are from a randomized controlled trial, specify `ps.model = ~1` as an intercept-only model.
`ps.method`	A character value for the method to estimate the propensity score. Allowed values include one of: `'glm'` for logistic regression with main effects only (default), or `'lasso'` for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in `ps.model`). Relevant only when `ps.model` has more than one variable.
`minPS`	A numerical value between 0 and 1 below which estimated propensity scores should be truncated. Default is `0.01`.
`maxPS`	A numerical value between 0 and 1 above which estimated propensity scores should be truncated. Must be strictly greater than `minPS`. Default is `0.99`.
`interactions`	A logical value indicating whether the outcome model should assume treatment-covariate interaction by `x`. If `TRUE`, interactions will be assumed only if at least 10 patients received each treatment option. Default is `TRUE`.
`n.boot`	A numeric value indicating the number of bootstrap samples used. Default is `500`.
`seed`	An optional integer specifying an initial randomization seed for reproducibility. Default is `NULL`, corresponding to no seed.

verbose An integer value indicating whether intermediate progress messages should be printed. 1 indicates messages are printed and 0 otherwise. Default is 0.

Details

This helper function estimates the average treatment effect (ATE) between two treatment groups in a given dataset. The ATE is estimated with a doubly robust estimator that accounts for imbalances in covariate distributions between the two treatment groups with inverse probability treatment weighting. For count outcomes, the estimated ATE is the estimated rate ratio between treatment 1 versus treatment 0.

Value

Return an item of the class atefit with the following elements:

- log.rate.ratio: A vector of numeric values of the estimated ATE (expressed as a log rate ratio of trt=1 over trt=0), the bootstrap standard error, the lower and upper limits of 95% confidence interval, and the p-value.
- rate0: A numeric value of the estimated rate in the group trt=0.
- rate1: A numeric value of the estimated rate in the group trt=1.
- trt.boot: Estimated log rate ratios in each bootstrap sample.
- warning: A warning message produced if the treatment variable was not coded as 0 or 1. The key to map the original coding of the variable to a 0-1 coding is displayed in the warning to facilitate the interpretation of the remaining of the output.

Examples

```
output <- atefitcount(data = countExample,
                      cate.model = y ~ age + female + previous_treatment +
                                   previous_cost + previous_number_relapses +
                                   offset(log(years)),
                      ps.model = trt ~ age + previous_treatment,
                      verbose = 1, n.boot = 50, seed = 999)
output
plot(output)
```

---

atefitmean *Doubly robust estimator of and inference for the average treatment effect for continuous data*

Description

Doubly robust estimator of the average treatment effect between two treatments, which is the rate ratio of treatment 1 over treatment 0 for count outcomes. Bootstrap is used for inference.

Usage

```
atefitmean(
  data,
  cate.model,
  ps.model,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  interactions = TRUE,
  n.boot = 500,
  plot.boot = FALSE,
  seed = NULL,
  verbose = 0
)
```

Arguments

data	A data frame containing the variables in the outcome and propensity score models; a data frame with n rows (1 row per observation).
cate.model	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
ps.model	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a RCT, specify ps.model as an intercept-only model.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value between 0 and 1 below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value between 0 and 1 above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
interactions	A logical value indicating whether the outcome model should be fitted separately by treatment arm with the variables in cate.model, which is equivalent to assuming treatment-covariate interaction by all of the variables in cate.model. If TRUE, the outcome model will be fitted separately by treatment arms only if at least 10 patients received each treatment option. Default is TRUE.
n.boot	A numeric value indicating the number of bootstrap samples used. Default is 500.
plot.boot	A logical value indicating whether histograms of the bootstrapped treatment effect estimates should be produced at every n.boot/10-th iteration and whether the final histogram should be outputted. Default is FALSE.
seed	An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.

verbose — An integer value indicating whether intermediate progress messages and histograms should be printed. 1 indicates messages are printed and 0 otherwise. Default is 0.

Details

This helper function estimates the average treatment effect (ATE) between two treatment groups in a given dataset specified by y, trt, x.cate, x.ps, time. The ATE is estimated with a doubly robust estimator that accounts for imbalances in covariate distributions between the two treatment groups with inverse probability treatment weighting. For count outcomes, the estimated ATE is the estimated rate ratio between treatment 1 versus treatment 0. Both original and log-transformed ATEs are returned, as well as the rate in either treatment group. If inference = TRUE, the variability of the estimated rate ratio is also calculated using bootstrap. Additional variability outputs include standard error of the log rate ratio, 95% confidence interval of the rate ratio, p-value, and a histogram of the log rate ratio.

Value

Return a list of 8 elements:

- log.rate.ratio: A numeric value of the estimated log rate ratio.
- se.boot.log.rate.ratio: A numeric value of the bootstrap standard error of log rate ratio.
- rate.ratio: A numeric value of the estimated rate ratio.
- rate.ratio0: A numeric value of the estimated rate in the group trt=0.
- rate.ratio1: A numeric value of the estimated rate in the group trt=1.
- rate.ratio.CIl: A numeric value of the lower limit 95% bootstrap confidence interval for estimated rate ratio.
- rate.ratio.CIu: A numeric value of the upper limit 95% bootstrap confidence interval for estimated rate ratio.
- pvalue: A numeric value of the p-value derived from the bootstrapped values based on a Chi-squared distribution.
- warning: A warning message produced if the treatment variable was not coded as 0/1. The key to map the original coding of the variable to a 0/1 key is displayed in the warning to facilitate the interpretation of the remaining of the output.
- plot: If plot.boot is TRUE, a histogram displaying the distribution of the bootstrapped log rate ratios. The red vertical reference line in the histogram represents the estimated log rate ratio.

Examples

```
# This module is not implemented yet!
```

`atefitsurv` *Doubly robust estimator of and inference for the average treatment effect for survival data*

Description

Doubly robust estimator of the average treatment effect between two treatments, which is the restricted mean time lost ratio for survival outcomes. Bootstrap is used for inference.

Usage

```
atefitsurv(
  data,
  cate.model,
  ps.model,
  ps.method = "glm",
  ipcw.model = NULL,
  ipcw.method = "breslow",
  minPS = 0.01,
  maxPS = 0.99,
  followup.time = NULL,
  tau0 = NULL,
  surv.min = 0.025,
  n.boot = 500,
  seed = NULL,
  verbose = 0
)
```

Arguments

`data`	A data frame containing the variables in the outcome, propensity score, and inverse probability of censoring models (if specified); a data frame with `n` rows (1 row per observation).
`cate.model`	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side. For survival outcomes, a `Surv` object must be used to describe the outcome.
`ps.model`	A formula describing the propensity score (PS) model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a randomized controlled trial, specify `ps.model = ~1` as an intercept-only model.
`ps.method`	A character value for the method to estimate the propensity score. Allowed values include one of: `'glm'` for logistic regression with main effects only (default), or `'lasso'` for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in `ps.model`). Relevant only when `ps.model` has more than one variable.

ipcw.model	A formula describing the inverse probability of censoring weighting (IPCW) model to be fitted. The left-hand side must be empty. Only applies for survival outcomes. Default is NULL, which corresponds to specifying the IPCW with the same covariates as the outcome model cate.model, plus the treatment.
ipcw.method	A character value for the censoring model. Only applies for survival outcomes. Allowed values are: 'breslow' (Cox regression with Breslow estimator of t he baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)' (accelerated failure time model with different distributions for y variable). Default is 'breslow'.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
followup.time	A column name in data specifying the maximum follow-up time, interpreted as the potential censoring time. Only applies for survival outcomes. Default is NULL, which corresponds to unknown potential censoring time.
tau0	The truncation time for defining restricted mean time lost. Only applies for survival outcomes. Default is NULL, which corresponds to setting the truncation time as the maximum survival time in the data.
surv.min	Lower truncation limit for the probability of being censored. It must be a positive value and should be chosen close to 0. Only applies for survival outcomes. Default is 0.025.
n.boot	A numeric value indicating the number of bootstrap samples used. Default is 500.
seed	An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.
verbose	An integer value indicating whether intermediate progress messages should be printed. 1 indicates messages are printed and 0 otherwise. Default is 0.

Details

This helper function estimates the average treatment effect (ATE) for survival data between two treatment groups in a given dataset. The ATE is estimated with a doubly robust estimator that accounts for imbalances in covariate distributions between the two treatment groups with inverse probability treatment and censoring weighting. For survival outcomes, the estimated ATE is the estimated by RMTL ratio between treatment 1 versus treatment 0. The log-transformed ATEs and log-transformed adjusted hazard ratios are returned, as well as the estimated RMST in either treatment group. The variability of the estimated RMTL ratio is calculated using bootstrap. Additional outputs include standard error of the log RMTL ratio, 95% confidence interval, p-value, and a histogram of the bootstrap estimates.

Value

Return an object of class atefit with 6 elements:

- rmst1: A vector of numeric values of the estimated RMST, bootstrap standard error, lower and upper limits of 95% confidence interval, and the p-value in the group trt=1.

- `rmst0`: A vector of numeric values of the estimated RMST, bootstrap standard error, lower and upper limits of 95% confidence interval, and the p-value in the group `trt=0`.
- `log.rmtl.ratio`: A vector of numeric values of the estimated log RMTL ratio of `trt=1` over `trt=0`, bootstrap standard error, lower and upper limits of 95% confidence interval, and the p-value.
- `log.hazard.ratio`: A vector of numeric values of the estimated adjusted log hazard ratio of `trt=1` over `trt=0`, bootstrap standard error, lower and upper limits of 95% confidence interval, and the p-value.
- `trt.boot`: Estimates of `rmst1`, `rmst0`, `log.rmtl.ratio` and `log.hazard.ratio` in each bootstrap sample.
- `warning`: A warning message produced if the treatment variable was not coded as 0/1. The key to map the original coding of the variable to a 0/1 key is displayed in the warning to facilitate the interpretation of the remaining of the output.

Examples

```
library(survival)
tau0 <- with(survivalExample,
             min(quantile(y[trt == "drug1"], 0.95), quantile(y[trt == "drug0"], 0.95)))

output <- atefitsurv(data = survivalExample,
                     cate.model = Surv(y, d) ~ age + female +
                                  previous_cost + previous_number_relapses,
                     ps.model = trt ~ age + previous_treatment,
                     tau0 = tau0,
                     n.boot = 50,
                     seed = 999,
                     verbose = 1)
output
plot(output)
```

---

auc

Compute the area under the curve using linear or natural spline interpolation

---

Description

This function computes the area under the curve for two vectors where one corresponds to the x values and the other corresponds to the y values. It supports both linear and spline interpolation.

Usage

```
auc(
  x,
  y,
```

```
  from = min(x, na.rm = TRUE),
  to = max(x, na.rm = TRUE),
  type = c("linear", "spline"),
  subdivisions = 100,
  ...
)
```

Arguments

x	A numeric vector of x values.
y	A numeric vector of y values of the same length as x.
from	The value from where to start calculating the area under the curve. Defaults to the smallest x value.
to	The value from where to end the calculation of the area under the curve. Defaults to the greatest x value.
type	The type of interpolation: "linear" or "spline". Defaults to "linear".
subdivisions	An integer indicating how many subdivisions to use for ‘integrate‘ (for spline-based approximations).
...	Additional arguments passed on to ‘approx‘ (for linear interpolations).

Value

A numeric value representing the area under the curve.

`balance.split` *Split the given dataset into balanced training and validation sets (within a pre-specified tolerance) Balanced means 1) The ratio of treated and controls is maintained in the training and validation sets 2) The covariate distributions are balanced between the training and validation sets*

Description

Split the given dataset into balanced training and validation sets (within a pre-specified tolerance) Balanced means 1) The ratio of treated and controls is maintained in the training and validation sets 2) The covariate distributions are balanced between the training and validation sets

Usage

```
balance.split(
  y,
  trt,
  x.cate,
  x.ps,
  time,
  minPS = 0.01,
```

```
  maxPS = 0.99,
  train.prop = 3/4,
  error.max = 0.1,
  max.iter = 5000
)
```

Arguments

y	Observed outcome; vector of size n (observations)
trt	Treatment received; vector of size n with treatment coded as 0/1
x.cate	Matrix of p.cate baseline covariates; dimension n by p.cate (covariates in the outcome model)
x.ps	Matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept); dimension n by p.ps + 1 (covariates in the propensity score model plus intercept)
time	Log-transformed person-years of follow-up; vector of size n
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
train.prop	A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is 3/4.
error.max	A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is 0.1.
max.iter	A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is 5,000.

Value

A list of 10 objects, 5 training and 5 validation of y, trt, x.cate, x.ps, time: y.train - observed outcome in the training set; vector of size m (observations in the training set) trt.train - treatment received in the training set; vector of size m coded as 0/1 x.cate.train - baseline covariates for the outcome model in the training set; matrix of dimension m by p.cate x.ps.train - baseline covariates (plus intercept) for the propensity score model in the training set; matrix of dimension m by p.ps + 1 time.train - log-transformed person-years of follow-up in the training set; vector of size m y.valid - observed outcome in the validation set; vector of size n-m trt.valid - treatment received in the validation set; vector of size n-m coded as 0/1 x.cate.valid - baseline covariates for the outcome model in the validation set; matrix of dimension n-m by p.cate x.ps.valid - baseline covariates (plus intercept) for the propensity score model in the validation set; matrix of dimension n-m by p.ps + 1 time.valid - log-transformed person-years of follow-up in the validation set; vector of size n-m

`balancemean.split` *Split the given dataset into balanced training and validation sets (within a pre-specified tolerance) Balanced means 1) The ratio of treated and controls is maintained in the training and validation sets 2) The covariate distributions are balanced between the training and validation sets*

Description

Split the given dataset into balanced training and validation sets (within a pre-specified tolerance) Balanced means 1) The ratio of treated and controls is maintained in the training and validation sets 2) The covariate distributions are balanced between the training and validation sets

Usage

```
balancemean.split(
  y,
  trt,
  x.cate,
  x.ps,
  minPS = 0.01,
  maxPS = 0.99,
  train.prop = 3/4,
  error.max = 0.1,
  max.iter = 5000
)
```

Arguments

`y`	Observed outcome; vector of size `n` (observations)
`trt`	Treatment received; vector of size `n` with treatment coded as 0/1
`x.cate`	Matrix of `p.cate` baseline covariates; dimension `n` by `p.cate` (covariates in the outcome model)
`x.ps`	Matrix of `p.ps` baseline covariates (plus a leading column of 1 for the intercept); dimension `n` by `p.ps + 1` (covariates in the propensity score model plus intercept)
`minPS`	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is `0.01`.
`maxPS`	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than `minPS`. Default is `0.99`.
`train.prop`	A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is `3/4`.
`error.max`	A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is `0.1`.

`max.iter` A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is `5,000`.

Value

A list of 10 objects, 5 training and 5 validation of y, trt, x.cate, x.ps, time: y.train - observed outcome in the training set; vector of size `m` (observations in the training set) trt.train - treatment received in the training set; vector of size `m` coded as 0/1 x.cate.train - baseline covariates for the outcome model in the training set; matrix of dimension `m` by `p.cate` x.ps.train - baseline covariates (plus intercept) for the propensity score model in the training set; matrix of dimension `m` by `p.ps + 1` y.valid - observed outcome in the validation set; vector of size `n-m` trt.valid - treatment received in the validation set; vector of size `n-m` coded as 0/1 x.cate.valid - baseline covariates for the outcome model in the validation set; matrix of dimension `n-m` by `p.cate` x.ps.valid - baseline covariates (plus intercept) for the propensity score model in the validation set; matrix of dimension `n-m` by `p.ps + 1` bestid.valid - id for the validation set by the best split; vector of size `n-m`

`balancesurv.split` *Split the given time-to-event dataset into balanced training and validation sets (within a pre-specified tolerance) Balanced means 1) The ratio of treated and controls is maintained in the training and validation sets 2) The covariate distributions are balanced between the training and validation sets*

Description

Split the given time-to-event dataset into balanced training and validation sets (within a pre-specified tolerance) Balanced means 1) The ratio of treated and controls is maintained in the training and validation sets 2) The covariate distributions are balanced between the training and validation sets

Usage

```
balancesurv.split(
  y,
  d,
  trt,
  x.cate,
  x.ps,
  x.ipcw,
  yf = NULL,
  train.prop = 3/4,
  error.max = 0.1,
  max.iter = 5000
)
```

Arguments

y	Observed survival or censoring time; vector of size n.
d	The event indicator, normally 1 = event, 0 = censored; vector of size n.
trt	Treatment received; vector of size n with treatment coded as 0/1.
x.cate	Matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate.
x.ps	Matrix of p.ps baseline covariates specified in the propensity score model; dimension n by p.ps.
x.ipcw	Matrix of p.ipw baseline covariate specified in inverse probability of censoring weighting; dimension n by p.ipw.
yf	Follow-up time, interpreted as the potential censoring time; vector of size n if the potential censoring time is known. If unknown, set yf == NULL and yf will be taken as y in the function.
train.prop	A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is 3/4.
error.max	A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is 0.1.
max.iter	A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is 5,000.

Value

A list of 14 objects, 7training and 7 validation of y, trt, x.cate, x.ps, x.ipcw, time, yf: y.train - observed survival or censoring time in the training set; vector of size m (observations in the training set) d.train - event indicator in the training set; vector of size m coded as 0/1 trt.train - treatment received in the training set; vector of size m coded as 0/1 x.cate.train - baseline covariates for the outcome model in the training set; matrix of dimension m by p.cate x.ps.train - baseline covariates (plus intercept) for the propensity score model in the training set; matrix of dimension m by p.ps + 1 x.ipcw.train - baseline covariates for inverse probability of censoring in the training set; matrix of dimension m by p.ipw yf.train - follow-up time in the training set; if known, vector of size m; if unknown, yf == NULL y.valid - observed survival or censoring time in the validation set; vector of size n-m d.valid - event indicator in the validation set; vector of size n-m coded as 0/1 trt.valid - treatment received in the validation set; vector of size n-m coded as 0/1 x.cate.valid - baseline covariates for the outcome model in the validation set; matrix of dimension n-m by p.cate x.ps.valid - baseline covariates (plus intercept) for the propensity score model in the validation set; matrix of dimension n-m by p.ps + 1 x.ipcw.valid - baseline covariates for inverse probability of censoring in the validation set; matrix of dimension n-m by p.ipw yf.valid - follow-up time in the training set; if known, vector of size n-m; if unknown, yf == NULL

`boxplot.precmed` *A set of box plots of estimated ATEs from the* `"precmed"` *object*

Description

Provides box plots which depict distributions of estimated ATEs for each multi-category subgroup in the validation set across all cross-validation iterations. The subgroups are mutually exclusive and are categorized by the CATE score percentiles (`prop.multi` specified in `catecv()` or `catecvmean()`). Box plots of mutually exclusive subgroups are constructed separately by scoring method specified in `catecv()`. This should be run only after results of `catecv()` or `catecvmean()`) have been obtained.

Usage

```
## S3 method for class 'precmed'
boxplot(
  x,
  ylab = NULL,
  plot.hr = FALSE,
  title = waiver(),
  theme = theme_classic(),
  ...
)
```

Arguments

`x`	An object of class `"precmed"`.
`ylab`	A character value for the y-axis label to describe what the ATE is. Default is `NULL`, which creates a default y-axis label based on available data.
`plot.hr`	A logical value indicating whether the hazard ratios should be plotted in the validation curves (`TRUE`). Otherwise, the restricted mean time lost is plotted (`FALSE`). This argument is only applicable to survival outcomes. Default is `FALSE`.
`title`	The text for the title
`theme`	Defaults to `theme_classic()`. Other options include `theme_grey()`, `theme_bw()`, `theme_light()`, `theme_dark()`, and `theme_void()`
`...`	Other parameters

Details

`boxplot()` takes in outputs from `catecv()` and generates the box plots of estimated ATEs for multi-category subgroups of the validation set. The box plots together with the overall ATE reference line can help compare the scoring methods' ability to distinguish subgroups of patients with different treatment effects.

For a given scoring method, box plots showing increasing or decreasing trends across the multi-category subgroups indicate presence of treatment effect heterogeneity (and the ability of the scoring method to capture it). On the contrary, box plots which are relatively aligned across the multi-category subgroups indicate absence of treatment effect heterogeneity (or the inability of the scoring method to capture it).

Value

Returns sets of box plots, one set for each scoring method, over each of the multi-category subgroups. A gray horizontal dashed line of the overall ATE is included as a reference.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

`plot` and `abc()` for `"precmed"` objects.

Examples

```
# Count outcome
eval_1 <- catecv(response = "count",
                 data = countExample,
                 score.method = "poisson",
                 cate.model = y ~ age + female + previous_treatment +
                                  previous_cost + previous_number_relapses +
                                  offset(log(years)),
                 ps.model = trt ~ age + previous_treatment,
                 higher.y = FALSE,
                 cv.n = 5)

boxplot(eval_1, ylab = "Rate ratio of drug1 vs drug0 in each subgroup")

# Survival outcome
library(survival)
tau0 <- with(survivalExample,
             min(quantile(y[trt == "drug1"], 0.95), quantile(y[trt == "drug0"], 0.95)))
eval_2 <- catecv(response = "survival",
                 data = survivalExample,
                 score.method = c("poisson", "randomForest"),
                 cate.model = Surv(y, d) ~ age + female + previous_cost +
                                           previous_number_relapses,
                 ps.model = trt ~ age + previous_treatment,
                 initial.predictor.method = "randomForest",
                 ipcw.model = ~ age + previous_cost + previous_treatment,
                 tau0 = tau0,
                 higher.y = TRUE,
                 cv.n = 5,
                 seed = 999)
```

```
boxplot(eval_2, ylab = "RMTL ratio of drug1 vs drug0 in each subgroup")
```

`catecv` *Cross-validation of the conditional average treatment effect (CATE) score for count, survival or continuous outcomes*

Description

Provides (doubly robust) estimation of the average treatment effect (ATE) for count, survival or continuous outcomes in nested and mutually exclusive subgroups of patients defined by an estimated conditional average treatment effect (CATE) score via cross-validation (CV).

Usage

```
catecv(
  response,
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  init.model = NULL,
  initial.predictor.method = NULL,
  ipcw.model = NULL,
  ipcw.method = "breslow",
  minPS = 0.01,
  maxPS = 0.99,
  followup.time = NULL,
  tau0 = NULL,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  prop.multi = c(0, 1/3, 2/3, 1),
  abc = TRUE,
  train.prop = 3/4,
  cv.n = 10,
  error.max = 0.1,
  max.iter = 5000,
  surv.min = 0.025,
  xvar.smooth.score = NULL,
  xvar.smooth.init = NULL,
  tree.depth = 2,
  n.trees.rf = 1000,
  n.trees.boosting = 200,
  B = 3,
```

```
  Kfold = 5,
  error.maxNR = 0.001,
  max.iterNR = 150,
  tune = c(0.5, 2),
  seed = NULL,
  plot.gbmperf = TRUE,
  verbose = 0
)
```

Arguments

response	A string describing the type of outcome in the data. Allowed values include "count" (see catecvcount()), "survival" (see catecvsurv()) and "continuous" (see catecvmean()) .
data	A data frame containing the variables in the outcome, propensity score, and inverse probability of censoring models (if specified); a data frame with n rows (1 row per observation).
score.method	A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'boosting', 'twoReg', 'contrastReg', 'poisson' (count and survival outcomes only), 'randomForest' (survival, continuous outcomes only), negBin (count outcomes only), 'gam' (continuous outcomes only), 'gaussian' (continuous outcomes only).
cate.model	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side. For survival outcomes, a Surv object must be used to describe the outcome.
ps.model	A formula describing the propensity score (PS) model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a randomized controlled trial, specify ps.model = ~1 as an intercept-only model.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
init.model	A formula describing the initial predictor model. The outcome must appear on the left-hand side. It must be specified when score.method = contrastReg or twoReg.
initial.predictor.method	A character vector for the method used to get initial outcome predictions conditional on the covariates specified in cate.model. Only applies when score.method includes 'twoReg' or 'contrastReg'. Allowed values include one of 'randomForest' (survival outcomes only), 'boosting', 'logistic' (survival outcomes only, fast), 'poisson' (count outcomes only, fast), 'gaussian' (continuous outcomes only) and 'gam' (count and continuous outcomes only). Default is NULL, which assigns 'boosting' for count outcomes and 'randomForest' for survival outcomes.

`ipcw.model` A formula describing the inverse probability of censoring weighting (IPCW) model to be fitted. The left-hand side must be empty. Only applies for survival outcomes. Default is `NULL`, which corresponds to specifying the IPCW with the same covariates as the outcome model `cate.model`, plus the treatment.

`ipcw.method` A character value for the censoring model. Only applies for survival outcomes. Allowed values are: `'breslow'` (Cox regression with Breslow estimator of t he baseline survivor function), `'aft (exponential)'`, `'aft (weibull)'`, `'aft (lognormal)'` or `'aft (loglogistic)'` (accelerated failure time model with different distributions for y variable). Default is `'breslow'`.

`minPS` A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is `0.01`.

`maxPS` A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than `minPS`. Default is `0.99`.

`followup.time` A column name in `data` specifying the maximum follow-up time, interpreted as the potential censoring time. Only applies for survival outcomes. Default is `NULL`, which corresponds to unknown potential censoring time.

`tau0` The truncation time for defining restricted mean time lost. Only applies for survival outcomes. Default is `NULL`, which corresponds to setting the truncation time as the maximum survival time in the data.

`higher.y` A logical value indicating whether higher (`TRUE`) or lower (`FALSE`) values of the outcome are more desirable. Default is `TRUE`.

`prop.cutoff` A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cut-off to separate observations in nested subgroups (below vs above cutoff). The length of `prop.cutoff` is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is `seq(0.5, 1, length = 6)`.

`prop.multi` A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of `length(prop.multi) > 2`. Each element represents the cutoff to separate the observations into `length(prop.multi) - 1` mutually exclusive subgroups. Default is `c(0, 1/3, 2/3, 1)`.

`abc` A logical value indicating whether the area between curves (ABC) should be calculated at each cross-validation iterations, for each `score.method`. Default is `TRUE`.

`train.prop` A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is `3/4`.

`cv.n` A positive integer value indicating the number of cross-validation iterations. Default is `10`.

`error.max` A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is `0.1`.

`max.iter` A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is `5,000`.

surv.min — Lower truncation limit for the probability of being censored. It must be a positive value and should be chosen close to 0. Only applies for survival outcomes. Default is 0.025.

xvar.smooth.score — A vector of characters indicating the name of the variables used as the smooth terms if score.method = 'gam'. The variables must be selected from the variables listed in cate.model.

xvar.smooth.init — A vector of characters indicating the name of the variables used as the smooth terms if initial.predictor.method = 'gam'. The variables must be selected from the variables listed in init.model. Default is NULL, which uses all variables in init.model.

tree.depth — A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 2.

n.trees.rf — A positive integer specifying the maximum number of trees in random forest. Used if score.method = 'ranfomForest' or if initial.predictor.method = 'randomForest' with score.method = 'twoReg' or 'contrastReg'. Only applies for survival outcomes. Default is 1000.

n.trees.boosting — A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 200.

B — A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold — A positive integer specifying the number of folds used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 5.

error.maxNR — A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

max.iterNR — A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.

tune — A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

seed — An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.

plot.gbmperf — A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

verbose — An integer value indicating what kind of intermediate progress messages should be printed. 0 means no outputs. 1 means only progress bar and run time. 2 means progress bar, run time, and all errors and warnings. Default is 0.

Details

For count response, see details in catecvcount(). For survival response, see details in catecvsurv(). For continuous response, see details in catecvmean().

Value

For count response, see description of outputs in catecvcount(). For survival response, see description of outputs in catecvsurv(). For continuous response, see description of outputs in catecvmean().

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

catefit() function and boxplot(), abc methods for "precmed" objects.

Examples

```
cate_1 <- catecv(response = "count",
                 data = countExample,
                 score.method = "poisson",
                 cate.model = y ~ age + female + previous_treatment +
                              previous_cost + previous_number_relapses +
                              offset(log(years)),
                 ps.model = trt ~ age + previous_treatment,
                 higher.y = FALSE, cv.n = 5, seed = 999, verbose = 1)

plot(cate_1, ylab = "RMTL ratio of drug1 vs drug0 in each subgroup")
boxplot(cate_1, ylab = "RMTL ratio of drug1 vs drug0 in each subgroup")
abc(cate_1)

# Survival outcome
library(survival)
tau0 <- with(survivalExample,
             min(quantile(y[trt == "drug1"], 0.95), quantile(y[trt == "drug0"], 0.95)))

cate_2 <- catecv(response = "survival",
                 data = survivalExample,
                 score.method = c("poisson", "randomForest"),
                 cate.model = Surv(y, d) ~ age + female + previous_cost +
                              previous_number_relapses,
                 ps.model = trt ~ age + previous_treatment,
                 initial.predictor.method = "randomForest",
```

```
                  ipcw.model = ~ age + previous_cost + previous_treatment,
                  tau0 = tau0,
                  higher.y = TRUE,
                  surv.min = 0.025,
                  cv.n = 5,
                  seed = 999)

plot(cate_2, ylab = "RMTL ratio of drug1 vs drug0 in each subgroup")
boxplot(cate_2, ylab = "RMTL ratio of drug1 vs drug0 in each subgroup")
abc(cate_2)
```

`catecvcount` *Cross-validation of the conditional average treatment effect (CATE) score for count outcomes*

Description

Provides doubly robust estimation of the average treatment effect (ATE) in nested and mutually exclusive subgroups of patients defined by an estimated conditional average treatment effect (CATE) score via cross-validation (CV). The CATE score can be estimated with up to 5 methods among the following: Poisson regression, boosting, two regressions, contrast regression, and negative binomial (see `score.method`).

Usage

```
catecvcount(
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  initial.predictor.method = "boosting",
  minPS = 0.01,
  maxPS = 0.99,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  prop.multi = c(0, 1/3, 2/3, 1),
  abc = TRUE,
  train.prop = 3/4,
  cv.n = 10,
  error.max = 0.1,
  max.iter = 5000,
  xvar.smooth = NULL,
  tree.depth = 2,
  n.trees.boosting = 200,
```

```
  B = 3,
  Kfold = 5,
  error.maxNR = 0.001,
  max.iterNR = 150,
  tune = c(0.5, 2),
  seed = NULL,
  plot.gbmperf = TRUE,
  verbose = 0,
  ...
)
```

Arguments

data	A data frame containing the variables in the outcome and propensity score model; a data frame with n rows (1 row per observation).
score.method	A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'boosting', 'poisson', 'twoReg', 'contrastReg', and 'negBin'.
cate.model	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
ps.model	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0 or 1. If data are from a randomized trial, specify ps.model as an intercept-only model.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
initial.predictor.method	A character vector for the method used to get initial outcome predictions conditional on the covariates in cate.model. Only applies when score.method includes 'twoReg' or 'contrastReg'. Allowed values include one of 'poisson' (fastest), 'boosting' and 'gam'. Default is 'boosting'.
minPS	A numerical value between 0 and 1 below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value between 0 and 1 above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
higher.y	A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.
prop.cutoff	A vector of numerical values between 0 and 1 specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).

prop.multi
: A vector of numerical values between 0 and 1 specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of length(prop.multi) > 2. Each element represents the cutoff to separate the observations into length(prop.multi) - 1 mutually exclusive subgroups. Default is c(0, 1/3, 2/3, 1).

abc
: A logical value indicating whether the area between curves (ABC) should be calculated at each cross-validation iterations, for each score.method. Default is TRUE.

train.prop
: A numerical value between 0 and 1 indicating the proportion of total data used for training. Default is 3/4.

cv.n
: A positive integer value indicating the number of cross-validation iterations. Default is 10.

error.max
: A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is 0.1.

max.iter
: A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is 5,000.

xvar.smooth
: A vector of characters indicating the name of the variables used as the smooth terms if initial.predictor.method = 'gam'. The variables must be selected from the variables listed in cate.model. Default is NULL, which uses all variables in cate.model.

tree.depth
: A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 2.

n.trees.boosting
: A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 200.

B
: A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold
: A positive integer specifying the number of folds used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 5.

error.maxNR
: A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

max.iterNR
: A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.

tune
: A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

seed	An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.
plot.gbmperf	A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.
verbose	An integer value indicating what kind of intermediate progress messages should be printed. 0 means no outputs. 1 means only progress bar and run time. 2 means progress bar, run time, and all errors and warnings. Default is 0.
...	Additional arguments for gbm()

Details

The CATE score represents an individual-level treatment effect expressed as a rate ratio for count outcomes. It can be estimated with boosting, Poisson regression, negative binomial regression, and the doubly robust estimator two regressions (Yadlowsky, 2020) applied separately by treatment group or with the other doubly robust estimator contrast regression (Yadlowsky, 2020) applied to the entire data set.

Internal CV is applied to reduce optimism in choosing the CATE estimation method that captures the most treatment effect heterogeneity. The CV is applied by repeating the following steps cv.n times:

1. Split the data into a training and validation set according to train.prop. The training and validation sets must be balanced with respect to covariate distributions and doubly robust rate ratio estimates (see error.max).
2. Estimate the CATE score in the training set with the specified scoring method.
3. Predict the CATE score in the validation set using the scoring model fitted from the training set.
4. Build nested subgroups of treatment responders in the training and validation sets, separately, and estimate the ATE within each nested subgroup. For each element i of prop.cutoff (e.g., prop.cutoff[i] = 0.6), take the following steps:
   (a) Identify high responders as observations with the 60% (i.e., prop.cutoff[i]x100%) highest (if higher.y = TRUE) or lowest (if higher.y = FALSE) estimated CATE scores.
   (b) Estimate the ATE in the subgroup of high responders using a doubly robust estimator.
   (c) Conversely, identify low responders as observations with the 40% (i.e., 1 - prop.cutoff[i]x100%) lowest (if higher.y = TRUE) or highest (if higher.y = FALSE) estimated CATE scores.
   (d) Estimate the ATE in the subgroup of low responders using a doubly robust estimator.
5. If abc = TRUE, calculate the area between the ATE and the series of ATEs in nested subgroups of high responders in the validation set.
6. Build mutually exclusive subgroups of treatment responders in the training and validation sets, separately, and estimate the ATE within each subgroup. Mutually exclusive subgroups are built by splitting the estimated CATE scores according to prop.multi.

Value

Returns a list containing the following components saved as a "precmed" object:

- ate.poisson: A list of results output if score.method includes 'poisson':
  - ate.est.train.high.cv: A matrix of numerical values with length(prop.cutoff) rows and cv.n columns. The ith row/jth column cell contains the estimated ATE in the nested subgroup of high responders defined by CATE score above (if higher.y = TRUE) or below (if higher.y = FALSE) the prop.cutoff[i]x100% percentile of the estimated CATE score in the training set in the jth cross-validation iteration.
  - ate.est.train.low.cv: A matrix of numerical values with length(prop.cutoff) - 1 rows and cv.n columns. The ith row/jth column cell contains the estimated ATE in the nested subgroup of low responders defined by CATE score below (if higher.y = TRUE) or above (if higher.y = FALSE) the prop.cutoff[i]x100% percentile of the estimated CATE score in the training set in the jth cross-validation iteration.
  - ate.est.valid.high.cv: Same as ate.est.train.high.cv, but in the validation set.
  - ate.est.valid.low.cv: Same as ate.est.train.low.cv, but in the validation set.
  - ate.est.train.group.cv: A matrix of numerical values with length(prop.multi) - 1 rows and cv.n columns. The jth column contains the estimated ATE in length(prop.multi) - 1 mutually exclusive subgroups defined by prop.multi in the training set in jth cross-validation iteration.
  - ate.est.valid.group.cv: Same as ate.est.train.group.cv, but in the validation set.
  - abc.valid: A vector of numerical values of length cv.n. The ith element returns the ABC of the validation curve in the ith cross-validation iteration. Only returned if abc = TRUE.
- ate.boosting: A list of results similar to ate.poisson output if score.method includes 'boosting'.
- ate.twoReg: A list of results similar to ate.poisson output if score.method includes 'twoReg'.
- ate.contrastReg: A list of results similar to ate.poisson output if score.method includes 'contrastReg'. This method has an additional element in the list of results:
  - converge.contrastReg.cv: A vector of logical value of length cv.n. The ith element indicates whether the algorithm converged in the ith cross-validation iteration.
- ate.negBin: A list of results similar to ate.poisson output if score.method includes 'negBin'.
- props: A list of 3 elements:
  - prop.onlyhigh: The original argument prop.cutoff, reformatted as necessary.
  - prop.bi: The original argument prop.cutoff, similar to prop.onlyhigh but reformatted to exclude 1.
  - prop.multi: The original argument prop.multi, reformatted as necessary to include 0 and 1.
- overall.ate.valid: A vector of numerical values of length cv.n. The ith element contains the ATE in the validation set of the ith cross-validation iteration, estimated with the doubly robust estimator.
- overall.ate.train: A vector of numerical values of length cv.n. The ith element contains the ATE in the training set of the ith cross-validation iteration, estimated with the doubly robust estimator.

- fgam: The formula used in GAM if initial.predictor.method = 'gam'.
- higher.y: The original higher.y argument.
- abc: The original abc argument.
- cv.n: The original cv.n argument.
- response: The type of response. Always 'count' for this function.
- formulas:A list of 3 elements: (1) cate.model argument, (2) ps.model argument and (3) original labels of the left-hand side variable in ps.model (treatment) if it was not 0/1.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.*. DOI: 10.1080/01621459.2020.1772080.

See Also

plot.precmed(), boxplot.precmed(), abc() methods for "precmed" objects, and catefitcount() function.

Examples

```
catecv <- catecvcount(data = countExample,
                      score.method = "poisson",
                      cate.model = y ~ age + female + previous_treatment +
                                   previous_cost + previous_number_relapses +
                                   offset(log(years)),
                      ps.model = trt ~ age + previous_treatment,
                      higher.y = FALSE,
                      cv.n = 5,
                      seed = 999,
                      plot.gbmperf = FALSE,
                      verbose = 1)

plot(catecv, ylab = "Rate ratio of drug1 vs drug0 in each subgroup")
boxplot(catecv, ylab = "Rate ratio of drug1 vs drug0 in each subgroup")
abc(catecv)
```

---

catecvmean *Cross-validation of the conditional average treatment effect (CATE) score for continuous outcomes*

Description

Provides doubly robust estimation of the average treatment effect (ATE) in nested and mutually exclusive subgroups of patients defined by an estimated conditional average treatment effect (CATE) score via cross-validation (CV). The CATE score can be estimated with up to 6 methods among the following: Linear regression, boosting, two regressions, contrast regression, random forest and generalized additive model (see score.method).

Usage

```
catecvmean(
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  init.model = NULL,
  initial.predictor.method = "boosting",
  minPS = 0.01,
  maxPS = 0.99,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  prop.multi = c(0, 1/3, 2/3, 1),
  abc = TRUE,
  train.prop = 3/4,
  cv.n = 10,
  error.max = 0.1,
  max.iter = 5000,
  xvar.smooth.score = NULL,
  xvar.smooth.init = NULL,
  tree.depth = 2,
  n.trees.rf = 1000,
  n.trees.boosting = 200,
  B = 3,
  Kfold = 6,
  plot.gbmperf = TRUE,
  error.maxNR = 0.001,
  tune = c(0.5, 2),
  seed = NULL,
  verbose = 0,
  ...
)
```

Arguments

`data`	A data frame containing the variables in the outcome and propensity score models; a data frame with `n` rows (1 row per observation).
`score.method`	A vector of one or multiple methods to estimate the CATE score. Allowed values are: `'boosting'`, `'gaussian'`, `'twoReg'`, `'contrastReg'`, `'randomForest'`, `'gam'`.
`cate.model`	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
`ps.model`	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a RCT, specify `ps.model` as an intercept-only model.

`ps.method` A character value for the method to estimate the propensity score. Allowed values include one of: `'glm'` for logistic regression with main effects only (default), or `'lasso'` for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in `ps.model`). Relevant only when `ps.model` has more than one variable.

`init.model` A formula describing the initial predictor model. The outcome must appear on the left-hand side. It must be specified when `score.method = contrastReg` or `twoReg`.

`initial.predictor.method` A character vector for the method used to get initial outcome predictions conditional on the covariates in `cate.model` in `score.method = 'twoReg'` and `'contrastReg'`. Allowed values include one of `'poisson'` (fastest), `'boosting'` and `'gam'`. Default is `'boosting'`.

`minPS` A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is `0.01`.

`maxPS` A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than `minPS`. Default is `0.99`.

`higher.y` A logical value indicating whether higher (`TRUE`) or lower (`FALSE`) values of the outcome are more desirable. Default is `TRUE`.

`prop.cutoff` A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of `prop.cutoff` is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is `seq(0.5, 1, length = 6)`.

`prop.multi` A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of `length(prop.multi) > 2`. Each element represents the cutoff to separate the observations into `length(prop.multi) - 1` mutually exclusive subgroups. Default is `c(0, 1/3, 2/3, 1)`.

`abc` A logical value indicating whether the area between curves (ABC) should be calculated at each cross-validation iterations, for each `score.method`. Default is `TRUE`.

`train.prop` A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is `3/4`.

`cv.n` A positive integer value indicating the number of cross-validation iterations. Default is `10`.

`error.max` A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is `0.1`.

`max.iter` A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is `5,000`.

xvar.smooth.score
: A vector of characters indicating the name of the variables used as the smooth terms if score.method = 'gam'. The variables must be selected from the variables listed in cate.model. Default is NULL, which uses all variables in cate.model.

xvar.smooth.init
: A vector of characters indicating the name of the variables used as the smooth terms if initial.predictor.method = 'gam'. The variables must be selected from the variables listed in init.model. Default is NULL, which uses all variables in init.model.

tree.depth
: A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 2.

n.trees.rf
: A positive integer specifying the maximum number of trees in random forest. Used if score.method = 'ranfomForest' or if initial.predictor.method = 'randomForest' with score.method = 'twoReg' or 'contrastReg'. Only applies for survival outcomes. Default is 1000.

n.trees.boosting
: A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 200.

B
: A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold
: A positive integer specifying the number of folds (parts) used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 6.

plot.gbmperf
: A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

error.maxNR
: A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

tune
: A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

seed
: An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.

verbose
: An integer value indicating what kind of intermediate progress messages should be printed. 0 means no outputs. 1 means only progress bar and run time. 2 means progress bar, run time, and all errors and warnings. Default is 0.

...
: Additional arguments for gbm()

Details

The CATE score represents an individual-level treatment effect for continuous data, estimated with boosting, linear regression, random forest, generalized additive model and the doubly robust estimator (two regressions, Yadlowsky, 2020) applied separately by treatment group or with the other doubly robust estimators (contrast regression, Yadlowsky, 2020) applied to the entire data set.

Internal CV is applied to reduce optimism in choosing the CATE estimation method that captures the most treatment effect heterogeneity. The CV is applied by repeating the following steps `cv.n` times:

1. Split the data into a training and validation set according to `train.prop`. The training and validation sets must be balanced with respect to covariate distributions and doubly robust rate ratio estimates (see `error.max`).
2. Estimate the CATE score in the training set with the specified scoring method.
3. Predict the CATE score in the validation set using the scoring model fitted from the training set.
4. Build nested subgroups of treatment responders in the training and validation sets, separately, and estimate the ATE within each nested subgroup. For each element i of `prop.cutoff` (e.g., `prop.cutoff[i]` = 0.6), take the following steps:
   (a) Identify high responders as observations with the 60% (i.e., `prop.cutoff[i]`x100%) highest (if `higher.y` = `TRUE`) or lowest (if `higher.y` = `FALSE`) estimated CATE scores.
   (b) Estimate the ATE in the subgroup of high responders using a doubly robust estimator.
   (c) Conversely, identify low responders as observations with the 40% (i.e., 1 - `prop.cutoff[i]`x100%) lowest (if `higher.y` = TRUE) or highest (if `higher.y` = FALSE) estimated CATE scores.
   (d) Estimate the ATE in the subgroup of low responders using a doubly robust estimator.
5. Build mutually exclusive subgroups of treatment responders in the training and validation sets, separately, and estimate the ATE within each subgroup. Mutually exclusive subgroups are built by splitting the estimated CATE scores according to `prop.multi`.
6. If `abc` = TRUE, calculate the area between the ATE and the series of ATEs in nested subgroups of high responders in the validation set.

Value

Returns a list containing the following components saved as a `"precmed"` object:

- `ate.gaussian`: A list of results output if `score.method` includes `'gaussian'`:
  - `ate.est.train.high.cv`: A matrix of numerical values with `length(prop.cutoff)` rows and `cv.n` columns. The ith column/jth row cell contains the estimated ATE in the nested subgroup of high responders defined by CATE score above (if `higher.y` = `TRUE`) or below (if `higher.y` = `FALSE`) the `prop.cutoff[j]`x100% percentile of the estimated CATE score in the training set in the ith cross-validation iteration.
  - `ate.est.train.low.cv`: A matrix of numerical values with `length(prop.cutoff) - 1` rows and `cv.n` columns. The ith column/jth row cell contains the estimated ATE in the nested subgroup of low responders defined by CATE score below (if `higher.y` = `TRUE`) or above (if `higher.y` = `FALSE`) the `prop.cutoff[j]`x100% percentile of the estimated CATE score in the training set in the ith cross-validation iteration.
  - `ate.est.valid.high.cv`: Same as `ate.est.train.high.cv`, but in the validation set.

  - ate.est.valid.low.cv: Same as ate.est.train.low.cv, but in the validation set.
  - ate.est.train.group.cv: A matrix of numerical values with length(prop.multi) - 1 rows and cv.n columns. The ith column contains the estimated ATE in length(prop.multi) - 1 mutually exclusive subgroups defined by prop.multi in the training set in ith cross-validation iteration.
  - ate.est.valid.group.cv: Same as ate.est.train.group.cv, but in the validation set.
  - abc.valid: A vector of numerical values of length cv.n, The ith element returns the ABC of the validation curve in the ith cross-validation iteration. Only returned if abc = TRUE.

- ate.boosting: A list of results similar to ate.gaussian output if score.method includes 'boosting'.
- ate.twoReg: A list of results similar to ate.gaussian output if score.method includes 'twoReg'.
- ate.contrastReg: A list of results similar to ate.gaussian output if score.method includes 'contrastReg'.
- ate.randomForest: A list of ATE output measured by the RMTL ratio if score.method includes 'randomForest':
- ate.gam: A list of results similar to ate.gaussian output if score.method includes 'gam'.
- props: A list of 3 elements:
  - prop.onlyhigh: The original argument prop.cutoff, reformatted as necessary.
  - prop.bi: The original argument prop.cutoff, similar to prop.onlyhigh but reformatted to exclude 1.
  - prop.multi: The original argument prop.multi, reformatted as necessary.
- overall.ate.train: A vector of numerical values of length cv.n. The ith element contains the ATE in the training set of the ith cross-validation iteration, estimated with the doubly robust estimator.
- overall.ate.valid: A vector of numerical values of length cv.n. The ith element contains the ATE in the validation set of the ith cross-validation iteration, estimated with the doubly robust estimator.
- higher.y: The original higher.y argument.
- abc: The original abc argument.
- cv.n: The original cv.n argument.
- response: The type of response. Always 'continuous' for this function.
- formulas:A list of 3 elements: (1) cate.model argument, (2) ps.model argument and (3) original labels of the left-hand side variable in ps.model (treatment) if it was not 0/1.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

plot.precmed(), boxplot.precmed(), abc() methods for "precmed" objects, and catefitmean() function.

Examples

```
# Not implemented yet!
```

---

catecvsurv *Cross-validation of the conditional average treatment effect (CATE) score for survival outcomes*

Description

Provides doubly robust estimation of the average treatment effect (ATE) by the RMTL (restricted mean time lost) ratio in nested and mutually exclusive subgroups of patients defined by an estimated conditional average treatment effect (CATE) score via cross-validation (CV). The CATE score can be estimated with up to 5 methods among the following: Random forest, boosting, poisson regression, two regressions, and contrast regression (see score.method).

Usage

```
catecvsurv(
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  initial.predictor.method = "randomForest",
  ipcw.model = NULL,
  ipcw.method = "breslow",
  minPS = 0.01,
  maxPS = 0.99,
  followup.time = NULL,
  tau0 = NULL,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  prop.multi = c(0, 1/3, 2/3, 1),
  abc = TRUE,
  train.prop = 3/4,
  cv.n = 10,
  error.max = 0.1,
  max.iter = 5000,
  surv.min = 0.025,
  tree.depth = 2,
  n.trees.rf = 1000,
```

```
  n.trees.boosting = 200,
  B = 3,
  Kfold = 5,
  error.maxNR = 0.001,
  max.iterNR = 150,
  tune = c(0.5, 2),
  seed = NULL,
  plot.gbmperf = TRUE,
  verbose = 0
)
```

Arguments

data	A data frame containing the variables in the outcome, propensity score, and inverse probability of censoring models (if specified); a data frame with n rows (1 row per observation).
score.method	A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'randomForest', 'boosting', 'poisson', 'twoReg', and 'contrastReg'.
cate.model	A standard Surv formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
ps.model	A formula describing the propensity score (PS) model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a randomized controlled trial, specify ps.model = ~1 as an intercept-only model.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
initial.predictor.method	A character vector for the method used to get initial outcome predictions conditional on the covariates specified in cate.model. Only applies when score.method includes 'twoReg' or 'contrastReg'. Allowed values include one of 'randomForest', 'boosting' and 'logistic' (fastest). Default is 'randomForest'.
ipcw.model	A formula describing the inverse probability of censoring weighting (IPCW) model to be fitted. The left-hand side must be empty. Default is ipcw.model = NULL, which corresponds to specifying the IPCW model with the same covariates as the outcome model cate.model plus the treatment.
ipcw.method	A character value for the censoring model. Allowed values are: 'breslow' (Cox regression with Breslow estimator of the baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)' (accelerated failure time model with different distributions for y variable). Default is 'breslow'.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

followup.time A column name in data specifying the maximum follow-up time, interpreted as the potential censoring time. Default is followup.time = NULL, which corresponds to unknown potential censoring time.

tau0 The truncation time for defining restricted mean time lost. Default is NULL, which corresponds to setting the truncation time as the maximum survival time in the data.

higher.y A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.

prop.cutoff A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).

prop.multi A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of length(prop.multi) > 2. Each element represents the cutoff to separate the observations into length(prop.multi) - 1 mutually exclusive subgroups. Default is c(0, 1/3, 2/3, 1).

abc A logical value indicating whether the area between curves (ABC) should be calculated at each cross-validation iterations, for each score.method. Default is TRUE.

train.prop A numerical value (in '(0, 1)') indicating the proportion of total data used for training. Default is 3/4.

cv.n A positive integer value indicating the number of cross-validation iterations. Default is 10.

error.max A numerical value > 0 indicating the tolerance (maximum value of error) for the largest standardized absolute difference in the covariate distributions or in the doubly robust estimated rate ratios between the training and validation sets. This is used to define a balanced training-validation splitting. Default is 0.1.

max.iter A positive integer value indicating the maximum number of iterations when searching for a balanced training-validation split. Default is 5,000.

surv.min Lower truncation limit for the probability of being censored. It must be a positive value and should be chosen close to 0. Default is 0.025.

tree.depth A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 2.

n.trees.rf A positive integer specifying the maximum number of trees in random forest. Used if score.method = 'ranfomForest' or if initial.predictor.method = 'randomForest' with score.method = 'twoReg' or 'contrastReg'. Only applies for survival outcomes. Default is 1000.

n.trees.boosting A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if score.method = 'boosting' or if initial.predictor.method

	= 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 200.
B	A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.
Kfold	A positive integer specifying the number of folds used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 5.
error.maxNR	A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.
max.iterNR	A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.
tune	A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).
seed	An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.
plot.gbmperf	A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.
verbose	An integer value indicating what kind of intermediate progress messages should be printed. 0 means no outputs. 1 means only progress bar and run time. 2 means progress bar, run time, and all errors and warnings. Default is 0.

Details

The CATE score represents an individual-level treatment effect expressed as the restricted mean survival time (RMTL) ratio) for survival outcomes. It can be estimated with boosting, Poisson regression, random forest, and the doubly robust estimator two regressions (Yadlowsky, 2020) applied separately by treatment group or with the other doubly robust estimator contrast regression (Yadlowsky, 2020) applied to the entire data set.

Internal CV is applied to reduce optimism in choosing the CATE estimation method that captures the most treatment effect heterogeneity. The CV is applied by repeating the following steps cv.n times:

1. Split the data into a training and validation set according to train.prop. The training and validation sets must be balanced with respect to covariate distributions and doubly robust RMTL ratio estimates (see error.max).
2. Estimate the CATE score in the training set with the specified scoring method.
3. Predict the CATE score in the validation set using the scoring model fitted from the training set.
4. Build nested subgroups of treatment responders in the training and validation sets, separately, and estimate the ATE within each nested subgroup. For each element i of prop.cutoff (e.g., prop.cutoff[i] = 0.6), take the following steps:

(a) Identify high responders as observations with the 60% (i.e., prop.cutoff[i]x100%) highest (if higher.y = FALSE) or lowest (if higher.y = TRUE) estimated CATE scores.

(b) Estimate the ATE in the subgroup of high responders using a doubly robust estimator.

(c) Conversely, identify low responders as observations with the 40% (i.e., 1 - prop.cutoff[i]x100%) lowest (if higher.y = FALSE) or highest (if higher.y = TRUE) estimated CATE scores.

(d) Estimate the ATE in the subgroup of low responders using a doubly robust estimator.

5. If abc = TRUE, calculate the area between the ATE and the series of ATEs in nested subgroups of high responders in the validation set.

6. Build mutually exclusive subgroups of treatment responders in the training and validation sets, separately, and estimate the ATE within each subgroup. Mutually exclusive subgroups are built by splitting the estimated CATE scores according to prop.multi.

Value

Returns a list containing the following components saved as a "precmed" object:

- ate.randomForest: A list of ATE output measured by the RMTL ratio if score.method includes 'randomForest':
  - ate.est.train.high.cv: A matrix of numerical values with length(prop.cutoff) rows and cv.n columns. The ith column/jth row cell contains the estimated ATE in the nested subgroup of high responders defined by CATE score above (if higher.y = FALSE) or below (if higher.y = TRUE) the prop.cutoff[j]x100% percentile of the estimated CATE score in the training set in the ith cross-validation iteration.
  - ate.est.train.low.cv: A matrix of numerical values with length(prop.cutoff) - 1 rows and cv.n columns. TThe ith column/jth row cell contains the estimated ATE in the nested subgroup of low responders defined by CATE score below (if higher.y = FALSE) or above (if higher.y = TRUE) the prop.cutoff[j]x100% percentile of the estimated CATE score in the training set in the ith cross-validation iteration.
  - ate.est.valid.high.cv: Same as ate.est.train.high.cv, but in the validation set.
  - ate.est.valid.low.cv: Same as ate.est.train.low.cv, but in the validation set.
  - ate.est.train.group.cv: A matrix of numerical values with length(prop.multi) - 1 rows and cv.n columns. The ith column contains the estimated ATE in length(prop.multi) - 1 mutually exclusive subgroups defined by prop.multi in the training set in ith cross-validation iteration.
  - ate.est.valid.group.cv: Same as ate.est.train.group.cv, but in the validation set.
  - abc.valid: A vector of numerical values of length cv.n, The ith element returns the ABC of the validation curve in the ith cross-validation iteration. Only returned if abc = TRUE.
- ate.boosting: A list of results similar to ate.randomForest output if score.method includes 'boosting'.
- ate.poisson: A list of results similar to ate.randomForest output if score.method includes 'poisson'.
- ate.twoReg: A list of results similar to ate.randomForest output if score.method includes 'twoReg'.

- ate.contrastReg: A list of results similar to ate.randomForest output if score.method includes 'contrastReg'. This method has an additional element in the list of results:
  - converge.contrastReg.cv: A vector of logical value of length cv.n. The ith element indicates whether the algorithm converged in the ith cross-validation iteration.
- hr.randomForest: A list of adjusted hazard ratio if score.method includes 'randomForest':
  - hr.est.train.high.cv: A matrix of numerical values with length(prop.cutoff) rows and cv.n columns. The ith column/jth row cell contains the estimated HR in the nested subgroup of high responders defined by CATE score above (if higher.y = FALSE) or below (if higher.y = TRUE) the prop.cutoff[j]x100% percentile of the estimated CATE score in the training set in the ith cross-validation iteration.
  - hr.est.train.low.cv: A matrix of numerical values with length(prop.cutoff) - 1 rows and cv.n columns. TThe ith column/jth row cell contains the estimated HR in the nested subgroup of low responders defined by CATE score below (if higher.y = FALSE) or above (if higher.y = TRUE) the prop.cutoff[j]x100% percentile of the estimated CATE score in the training set in the ith cross-validation iteration.
  - hr.est.valid.high.cv: Same as hr.est.train.high.cv, but in the validation set.
  - hr.est.valid.low.cv: Same as hr.est.train.low.cv, but in the validation set.
  - hr.est.train.group.cv: A matrix of numerical values with length(prop.multi) - 1 rows and cv.n columns. The ith column contains the estimated HR in length(prop.multi) - 1 mutually exclusive subgroups defined by prop.multi in the training set in ith cross-validation iteration.
  - hr.est.valid.group.cv: Same as hr.est.train.group.cv, but in the validation set.
- hr.boosting: A list of results similar to hr.randomForest output if score.method includes 'boosting'.
- hr.poisson: A list of results similar to hr.randomForest output if score.method includes 'poisson'.
- hr.twoReg: A list of results similar to hr.randomForest output if score.method includes 'twoReg'.
- hr.contrastReg: A list of results similar to hr.randomForest output if score.method includes 'contrastReg'.
- props: A list of 3 elements:
  - prop.onlyhigh: The original argument prop.cutoff, reformatted as necessary.
  - prop.bi: The original argument prop.cutoff, similar to prop.onlyhigh but reformatted to exclude 1.
  - prop.multi: The original argument prop.multi, reformatted as necessary to include 0 and 1.
- overall.ate.train: A vector of numerical values of length cv.n. The ith element contains the ATE (RMTL ratio) in the training set of the ith cross-validation iteration, estimated with the doubly robust estimator.
- overall.hr.train: A vector of numerical values of length cv.n. The ith element contains the ATE (HR) in the training set of the ith cross-validation iteration.
- overall.ate.valid: A vector of numerical values of length cv.n. The ith element contains the ATE (RMTL ratio) in the validation set of the ith cross-validation iteration, estimated with the doubly robust estimator.

- `overall.hr.valid`: A vector of numerical values of length `cv.n`. The ith element contains the ATE (HR) in the validation set of the ith cross-validation iteration.
- `errors/warnings`: A nested list of errors and warnings that were wrapped during the calculation of ATE. Errors and warnings are organized by `score.method` and position in the CV flow.
- `higher.y`: The original `higher.y` argument.
- `abc`: The original `abc` argument.
- `cv.n`: The original `cv.n` argument.
- `response`: The type of response. Always 'survival' for this function.
- `formulas`:A list of 3 elements: (1) `cate.model` argument, (2) `ps.model` argument and (3) original labels of the left-hand side variable in `ps.model` (treatment) if it was not 0/1.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.*. DOI: 10.1080/01621459.2020.1772080.

See Also

`catefitsurv()` function and `boxplot()`, `abc` methods for `"precmed"` objects.

Examples

```
library(survival)

tau0 <- with(survivalExample,
             min(quantile(y[trt == "drug1"], 0.95), quantile(y[trt == "drug0"], 0.95)))

catecv <- catecvsurv(data = survivalExample,
                     score.method = "poisson",
                     cate.model = Surv(y, d) ~ age + female + previous_cost +
                                               previous_number_relapses,
                     ps.model = trt ~ age + previous_treatment,
                     initial.predictor.method = "logistic",
                     ipcw.model = ~ age + previous_cost + previous_treatment,
                     tau0 = tau0,
                     higher.y = TRUE,
                     cv.n = 5, seed = 999, verbose = 1)

# Try:
plot(catecv, ylab = "RMTL ratio of drug1 vs drug0 in each subgroup")
boxplot(catecv, ylab = "RMTL ratio of drug1 vs drug0 in each subgroup")
abc(catecv)
```

`catefit` *Estimation of the conditional average treatment effect (CATE) score for count, survival and continuous data*

Description

Provides singly robust and doubly robust estimation of CATE score for count, survival and continuous data with up the following scoring methods among the following: Random forest (survival, continuous only), boosting, poisson regression (count, survival only), two regressions, contrast regression, negative binomial regression (count only), linear regression (continuous only), and generalized additive model (continuous only).

Usage

```
catefit(
  response,
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  init.model = NULL,
  initial.predictor.method = NULL,
  ipcw.model = NULL,
  ipcw.method = "breslow",
  minPS = 0.01,
  maxPS = 0.99,
  followup.time = NULL,
  tau0 = NULL,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  surv.min = 0.025,
  xvar.smooth.score = NULL,
  xvar.smooth.init = NULL,
  tree.depth = 2,
  n.trees.rf = 1000,
  n.trees.boosting = 200,
  B = 3,
  Kfold = 5,
  error.maxNR = 0.001,
  max.iterNR = 150,
  tune = c(0.5, 2),
  seed = NULL,
  plot.gbmperf = TRUE,
  verbose = 0,
  ...
)
```

Arguments

response	A string describing the type of outcome in the data. Allowed values include "count" (see catecvcount()), "survival" (see catecvsurv()) and "continuous" (see catecvmean()).
data	A data frame containing the variables in the outcome, propensity score, and inverse probability of censoring models (if specified); a data frame with n rows (1 row per observation).
score.method	A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'boosting', 'twoReg', 'contrastReg', 'poisson' (count and survival outcomes only), 'randomForest' (survival, continuous outcomes only), negBin (count outcomes only), 'gam' (continuous outcomes only), 'gaussian' (continuous outcomes only).
cate.model	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side. For survival outcomes, a Surv object must be used to describe the outcome.
ps.model	A formula describing the propensity score (PS) model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a randomized controlled trial, specify ps.model = ~1 as an intercept-only model.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
init.model	A formula describing the initial predictor model. The outcome must appear on the left-hand side. It must be specified when score.method = contrastReg or twoReg.
initial.predictor.method	A character vector for the method used to get initial outcome predictions conditional on the covariates specified in cate.model. Only applies when score.method includes 'twoReg' or 'contrastReg'.Allowed values include one of 'randomForest' (survival outcomes only), 'boosting', 'logistic' (survival outcomes only, fast), 'poisson' (count outcomes only, fast), 'gaussian' (continuous outcomes only) and 'gam' (count and continuous outcomes only). Default is NULL, which assigns 'boosting' for count outcomes and 'randomForest' for survival outcomes.
ipcw.model	A formula describing the inverse probability of censoring weighting (IPCW) model to be fitted. The left-hand side must be empty. Only applies for survival outcomes. Default is NULL, which corresponds to specifying the IPCW with the same covariates as the outcome model cate.model, plus the treatment.
ipcw.method	A character value for the censoring model. Only applies for survival outcomes. Allowed values are: 'breslow' (Cox regression with Breslow estimator of t he baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)' (accelerated failure time model with different distributions for y variable). Default is 'breslow'.

`minPS` A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is `0.01`.

`maxPS` A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than `minPS`. Default is `0.99`.

`followup.time` A column name in `data` specifying the maximum follow-up time, interpreted as the potential censoring time. Only applies for survival outcomes. Default is `NULL`, which corresponds to unknown potential censoring time.

`tau0` The truncation time for defining restricted mean time lost. Only applies for survival outcomes. Default is `NULL`, which corresponds to setting the truncation time as the maximum survival time in the data.

`higher.y` A logical value indicating whether higher (`TRUE`) or lower (`FALSE`) values of the outcome are more desirable. Default is `TRUE`.

`prop.cutoff` A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of `prop.cutoff` is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is `seq(0.5, 1, length = 6)`.

`surv.min` Lower truncation limit for the probability of being censored. It must be a positive value and should be chosen close to 0. Only applies for survival outcomes. Default is `0.025`.

`xvar.smooth.score`
A vector of characters indicating the name of the variables used as the smooth terms if `score.method = 'gam'`. The variables must be selected from the variables listed in `cate.model`.

`xvar.smooth.init`
A vector of characters indicating the name of the variables used as the smooth terms if `initial.predictor.method = 'gam'`. The variables must be selected from the variables listed in `init.model`. Default is `NULL`, which uses all variables in `init.model`.

`tree.depth` A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if `score.method = 'boosting'` or if `initial.predictor.method = 'boosting'` with `score.method = 'twoReg'` or `'contrastReg'`. Default is 2.

`n.trees.rf` A positive integer specifying the maximum number of trees in random forest. Used if `score.method = 'ranfomForest'` or if `initial.predictor.method = 'randomForest'` with `score.method = 'twoReg'` or `'contrastReg'`. Only applies for survival outcomes. Default is `1000`.

`n.trees.boosting`
A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if `score.method = 'boosting'` or if `initial.predictor.method = 'boosting'` with `score.method = 'twoReg'` or `'contrastReg'`. Default is `200`.

`B` A positive integer specifying the number of time cross-fitting is repeated in `score.method = 'twoReg'` and `'contrastReg'`. Default is `3`.

Kfold	A positive integer specifying the number of folds used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 5.
error.maxNR	A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.
max.iterNR	A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.
tune	A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).
seed	An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.
plot.gbmperf	A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.
verbose	An integer value indicating whether intermediate progress messages and histograms should be printed. 1 indicates messages are printed and 0 otherwise. Default is 0.
...	Additional arguments for gbm()

Details

For count response, see details in catefitcount(). For survival response, see details in catefitsurv().

Value

For count response, see description of outputs in catefitcount(). For survival response, see description of outputs in catefitsurv().

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

catecv()

Examples

```
# Count outcome
fit_1 <- catefit(response = "count",
                 data = countExample,
                 score.method = "poisson",
```

```
                  cate.model = y ~ age + female + previous_treatment +
                                   previous_cost + previous_number_relapses +
                                   offset(log(years)),
                  ps.model = trt ~ age + previous_treatment,
                  higher.y = TRUE,
                  seed = 999)

coef(fit_1)


# Survival outcome
library(survival)
tau0 <- with(survivalExample,
               min(quantile(y[trt == "drug1"], 0.95), quantile(y[trt == "drug0"], 0.95)))

fit_2 <- catefit(response = "survival",
                  data = survivalExample,
                  score.method = c("poisson", "boosting", "randomForest"),
                  cate.model = Surv(y, d) ~ age + female + previous_cost +
                                            previous_number_relapses,
                  ps.model = trt ~ age + previous_treatment,
                  initial.predictor.method = "logistic",
                  ipcw.model = ~ age + previous_cost + previous_treatment,
                  tau0 = tau0, higher.y = TRUE, seed = 999, n.cores = 1)

coef(fit_2)
```

`catefitcount` *Estimation of the conditional average treatment effect (CATE) score for count data*

Description

Provides singly robust and doubly robust estimation of CATE score with up to 5 scoring methods among the following: Poisson regression, boosting, two regressions, contrast regression, and negative binomial.

Usage

```
catefitcount(
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  initial.predictor.method = "boosting",
  minPS = 0.01,
```

```
  maxPS = 0.99,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  xvar.smooth = NULL,
  tree.depth = 2,
  n.trees.boosting = 200,
  B = 3,
  Kfold = 5,
  error.maxNR = 0.001,
  max.iterNR = 150,
  tune = c(0.5, 2),
  seed = NULL,
  plot.gbmperf = FALSE,
  verbose = 0,
  ...
)
```

Arguments

`data`	A data frame containing the variables in the outcome and propensity score model; a data frame with `n` rows (1 row per observation).
`score.method`	A vector of one or multiple methods to estimate the CATE score. Allowed values are: `'boosting'`, `'poisson'`, `'twoReg'`, `'contrastReg'`, and `'negBin'`.
`cate.model`	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
`ps.model`	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a randomized trial, specify `ps.model` as an intercept-only model.
`ps.method`	A character value for the method to estimate the propensity score. Allowed values include one of: `'glm'` for logistic regression with main effects only (default), or `'lasso'` for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in `ps.model`). Relevant only when `ps.model` has more than one variable.
`initial.predictor.method`	A character vector for the method used to get initial outcome predictions conditional on the covariates in `cate.model`. Only applies when `score.method` includes `'twoReg'` or `'contrastReg'`. Allowed values include one of `'poisson'` (fastest), `'boosting'` and `'gam'`. Default is `'boosting'`.
`minPS`	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is `0.01`.
`maxPS`	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than `minPS`. Default is `0.99`.
`higher.y`	A logical value indicating whether higher (`TRUE`) or lower (`FALSE`) values of the outcome are more desirable. Default is `TRUE`.

prop.cutoff A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).

xvar.smooth A vector of characters indicating the name of the variables used as the smooth terms if initial.predictor.method = 'gam'. The variables must be selected from the variables listed in cate.model. Default is NULL, which uses all variables in cate.model.

tree.depth A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 2.

n.trees.boosting
A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 200.

B A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold A positive integer specifying the number of folds used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 5.

error.maxNR A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

max.iterNR A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.

tune A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

seed An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.

plot.gbmperf A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

verbose An integer value indicating what kind of intermediate progress messages should be printed. 0 means no outputs. 1 means only progress and run time. 2 means progress, run time, and all errors and warnings. Default is 0.

... Additional arguments for gbm()

Details

The CATE score represents an individual-level treatment effect, estimated with either Poisson regression, boosting or negative binomial regression applied separately by treatment group or with two doubly robust estimators, two regressions and contrast regression (Yadlowsky, 2020) applied to the entire dataset.

`catefitcount()` provides the coefficients of the CATE score for each scoring method requested through `score.method`. Currently, contrast regression is the only method which allows for inference of the CATE coefficients by providing standard errors of the coefficients. The coefficients can be used to learn the effect size of each variable and predict the CATE score for a new observation.

`catefitcount()` also provides the predicted CATE score of each observation in the data set, for each scoring method. The predictions allow ranking the observations from potentially high responders to the treatment to potentially low or standard responders.

The estimated ATE among nested subgroups of high responders are also provided by scoring method. Note that the ATEs in `catefitcount()` are derived based on the CATE score which is estimated using the same data sample. Therefore, overfitting may be an issue. `catecvcount()` is more suitable to inspect the estimated ATEs across scoring methods as it implements internal cross validation to reduce optimism.

Value

Returns a list containing the following components:

- `ate.poisson`: A vector of numerical values of length `prop.cutoff` containing the estimated ATE in nested subgroups (defined by `prop.cutoff`) constructed based on the estimated CATE scores with poisson regression. Only provided if `score.method` includes `'poisson'`.
- `ate.boosting`: Same as `ate.poisson`, but with the nested subgroups based the estimated CATE scores with boosting. Only provided if `score.method` includes `'boosting'`.
- `ate.twoReg`: Same as `ate.poisson`, but with the nested subgroups based the estimated CATE scores with two regressions. Only provided if `score.method` includes `'twoReg'`.
- `ate.contrastReg`: Same as `ate.poisson`, but with the nested subgroups based the estimated CATE scores with contrast regression. Only provided if `score.method` includes `'contrastReg'`.
- `ate.negBin`: Same as `ate.poisson`, but with the nested subgroups based the estimated CATE scores with negative binomial regression. Only provided if `score.method` includes `'negBin'`.
- `score.poisson`: A vector of numerical values of length n (number of observations in `data`) containing the estimated log-CATE scores according to the Poisson regression. Only provided if `score.method` includes `'poisson'`.
- `score.boosting`: Same as `score.poisson`, but with estimated log-CATE score according to boosting. Only provided if `score.method` includes `'boosting'`.
- `score.twoReg`: Same as `score.poisson`, but with estimated log-CATE score according to two regressions. Only provided if `score.method` includes `'twoReg'`.
- `score.contrastReg`: Same as `score.poisson`, but with estimated log-CATE score according to contrast regression. Only provided if `score.method` includes `'contrastReg'`.
- `score.negBin`: Same as `score.poisson`, but with estimated log-CATE score according to negative binomial regression. Only provided if `score.method` includes `'negBin'`.
- `fit`: Additional details on model fitting if `score.method` includes 'boosting' or 'contrastReg':

  - result.boosting: Details on the boosting model fitted to observations with treatment = 0 ($fit0.boosting) and to observations with treatment = 1 ($fit1.boosting). Only provided if score.method includes 'boosting'.
  - result.contrastReg$sigma.contrastReg: Variance-covariance matrix of the estimated log-CATE coefficients in contrast regression. Only provided if score.method includes 'contrastReg'.

- coefficients: A data frame with the coefficients of the estimated log-CATE score by score.method. The data frame has number of rows equal to the number of covariates in cate.model and number of columns equal to length(score.method). If score.method includes 'contrastReg', the data frame has an additional column containing the standard errors of the coefficients estimated with contrast regression. 'boosting' does not have coefficient results because tree-based methods typically do not express the log-CATE as a linear combination of coefficients and covariates.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

catecvcount()

Examples

```
fit <- catefitcount(data = countExample,
                    score.method = "poisson",
                    cate.model = y ~ age + female + previous_treatment +
                                 previous_cost + previous_number_relapses +
                                 offset(log(years)),
                    ps.model = trt ~ age + previous_treatment,
                    higher.y = FALSE,
                    seed = 999, verbose = 1)
```

---

catefitmean *Estimation of the conditional average treatment effect (CATE) score for continuous data*

Description

Provides singly robust and doubly robust estimation of CATE score with up to 6 scoring methods among the following: Linear regression, boosting, two regressions, contrast regression, random forest and generalized additive model.

Usage

```
catefitmean(
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  init.model = NULL,
  initial.predictor.method = "boosting",
  minPS = 0.01,
  maxPS = 0.99,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  xvar.smooth.score = NULL,
  xvar.smooth.init = NULL,
  tree.depth = 2,
  n.trees.rf = 1000,
  n.trees.boosting = 200,
  B = 3,
  Kfold = 6,
  plot.gbmperf = FALSE,
  error.maxNR = 0.001,
  tune = c(0.5, 2),
  seed = NULL,
  verbose = 0,
  ...
)
```

Arguments

`data`	A data frame containing the variables in the outcome and propensity score models; a data frame with `n` rows (1 row per observation).
`score.method`	A vector of one or multiple methods to estimate the CATE score. Allowed values are: `'boosting'`, `'gaussian'`, `'twoReg'`, `'contrastReg'`, `'randomForest'`, `'gam'`.
`cate.model`	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
`ps.model`	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a RCT, specify `ps.model` as an intercept-only model.
`ps.method`	A character value for the method to estimate the propensity score. Allowed values include one of: `'glm'` for logistic regression with main effects only (default), or `'lasso'` for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if not specified in `ps.model`).
`init.model`	A formula describing the initial predictor model. The outcome must appear on the left-hand side. It must be specified when `score.method = contrastReg` or `twoReg`.

initial.predictor.method
: A character vector for the method used to get initial outcome predictions conditional on the covariates in init.model in score.method = 'twoReg' and 'contrastReg'. Allowed values include one of 'gaussian' (fastest), 'boosting' and 'gam'. Default is 'boosting'.

minPS
: A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.

maxPS
: A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

higher.y
: A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.

prop.cutoff
: A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).

xvar.smooth.score
: A vector of characters indicating the name of the variables used as the smooth terms if score.method = 'gam'. The variables must be selected from the variables listed in cate.model. Default is NULL, which uses all variables in cate.model.

xvar.smooth.init
: A vector of characters indicating the name of the variables used as the smooth terms if initial.predictor.method = 'gam'. The variables must be selected from the variables listed in init.model. Default is NULL, which uses all variables in init.model.

tree.depth
: A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 2.

n.trees.rf
: A positive integer specifying the number of trees. Used only if score.method = 'randomForest'. Default is 1000.

n.trees.boosting
: A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 200.

B
: A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold
: A positive integer specifying the number of folds (parts) used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 6.

plot.gbmperf
: A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

error.maxNR	A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.
tune	A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).
seed	An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.
verbose	An integer value indicating what kind of intermediate progress messages should be printed. 0 means no outputs. 1 means only progress and run time. 2 means progress, run time, and all errors and warnings. Default is 0.
...	Additional arguments for gbm()

Details

The CATE score represents an individual-level treatment effect, estimated with either linear regression, boosting, random forest and generalized additive model applied separately by treatment group or with two doubly robust estimators, two regressions and contrast regression (Yadlowsky, 2020) applied to the entire dataset.

catefitmean() provides the coefficients of the CATE score for each scoring method requested through score.method. Currently, contrast regression is the only method which allows for inference of the CATE coefficients by providing standard errors of the coefficients. The coefficients can be used to learn the effect size of each variable and predict the CATE score for a new observation.

catefitmean() also provides the predicted CATE score of each observation in the data set, for each scoring method. The predictions allow ranking the observations from potentially high responders to the treatment to potentially low or standard responders.

The estimated ATE among nested subgroups of high responders are also provided by scoring method. Note that the ATEs in catefitmean() are derived based on the CATE score which is estimated using the same data sample. Therefore, overfitting may be an issue. catefitmean() is more suitable to inspect the estimated ATEs across scoring methods as it implements internal cross validation to reduce optimism.

Value

Returns a list containing the following components:

- ate.gaussian: A vector of numerical values of length prop.cutoff containing the estimated ATE in nested subgroups (defined by prop.cutoff) constructed based on the estimated CATE scores with Poisson regression. Only provided if score.method includes 'gaussian'.
- ate.boosting: Same as ate.gaussian, but with the nested subgroups based the estimated CATE scores with boosting. Only provided if score.method includes 'boosting'.
- ate.twoReg: Same as ate.gaussian, but with the nested subgroups based the estimated CATE scores with two regressions. Only provided if score.method includes 'twoReg'.
- ate.contrastReg: Same as ate.gaussian, but with the nested subgroups based the estimated CATE scores with contrast regression. Only provided if score.method includes 'contrastReg'.

- `ate.randomForest`: Same as `ate.gaussian`, but with the nested subgroups based the estimated CATE scores with random forest. Only provided if `score.method` includes `'gam'`.
- `ate.gam`: Same as `ate.gaussian`, but with the nested subgroups based the estimated CATE scores with generalized additive model. Only provided if `score.method` includes `'gam'`.
- `score.gaussian`: A vector of numerical values of length n (number of observations in `data`) containing the estimated CATE scores according to the linear regression. Only provided if `score.method` includes `'gaussian'`.
- `score.boosting`: Same as `score.gaussian`, but with estimated CATE score according to boosting. Only provided if `score.method` includes `'boosting'`.
- `score.twoReg`: Same as `score.gaussian`, but with estimated CATE score according to two regressions. Only provided if `score.method` includes `'twoReg'`.
- `score.contrastReg`: Same as `score.gaussian`, but with estimated CATE score according to contrast regression. Only provided if `score.method` includes `'contrastReg'`.
- `score.randomForest`: Same as `score.gaussian`, but with estimated CATE score according to random forest. Only provided if `score.method` includes `'randomForest'`.
- `score.gam`: Same as `score.gaussian`, but with estimated CATE score according to generalized additive model. Only provided if `score.method` includes `'gam'`.
- `fit`: Additional details on model fitting if `score.method` includes 'boosting' or 'contrastReg':
  - `result.boosting`: Details on the boosting model fitted to observations with treatment = 0 (`$fit0.boosting`) and to observations with treatment = 1 (`$fit1.boosting`). Only provided if `score.method` includes `'boosting'`.
  - `result.randomForest`: Details on the boosting model fitted to observations with treatment = 0 (`$fit0.randomForest`) and to observations with treatment = 1 (`$fit1.randomForest`). Only provided if `score.method` includes `'randomForest'`.
  - `result.gam`: Details on the boosting model fitted to observations with treatment = 0 (`$fit0.gam`) and to observations with treatment = 1 (`$fit1.gam`). Only provided if `score.method` includes `'gam'`.
  - `result.contrastReg$sigma.contrastReg`: Variance-covariance matrix of the estimated CATE coefficients in contrast regression. Only provided if `score.method` includes `'contrastReg'`.
- `coefficients`: A data frame with the coefficients of the estimated CATE score by `score.method`. The data frame has number of rows equal to the number of covariates in `cate.model` and number of columns equal to `length(score.method)`. If `score.method` includes `'contrastReg'`, the data frame has an additional column containing the standard errors of the coefficients estimated with contrast regression. `'boosting'`, `'randomForest'`, `'gam'` do not have coefficient results because these methods do not express the CATE as a linear combination of coefficients and covariates.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

`catecvmean()` function

`catefitsurv` *Estimation of the conditional average treatment effect (CATE) score for survival data*

Description

Provides singly robust and doubly robust estimation of CATE score for survival data with up to 5 scoring methods among the following: Random forest, boosting, poisson regression, two regressions, and contrast regression.

Usage

```
catefitsurv(
  data,
  score.method,
  cate.model,
  ps.model,
  ps.method = "glm",
  initial.predictor.method = "randomForest",
  ipcw.model = NULL,
  ipcw.method = "breslow",
  minPS = 0.01,
  maxPS = 0.99,
  followup.time = NULL,
  tau0 = NULL,
  higher.y = TRUE,
  prop.cutoff = seq(0.5, 1, length = 6),
  surv.min = 0.025,
  tree.depth = 2,
  n.trees.rf = 1000,
  n.trees.boosting = 200,
  B = 3,
  Kfold = 5,
  plot.gbmperf = TRUE,
  error.maxNR = 0.001,
  max.iterNR = 100,
  tune = c(0.5, 2),
  seed = NULL,
  verbose = 0,
  ...
)
```

Arguments

`data`	A data frame containing the variables in the outcome, propensity score, and inverse probability of censoring models (if specified); a data frame with `n` rows (1 row per observation).

score.method A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'randomForest', 'boosting', 'poisson', 'twoReg', and 'contrastReg'.

cate.model A standard Surv formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.

ps.model A formula describing the propensity score (PS) model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a randomized controlled trial, specify ps.model = ~1 as an intercept-only model.

ps.method A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.

initial.predictor.method
A character vector for the method used to get initial outcome predictions conditional on the covariates specified in cate.model. Only applies when score.method includes 'twoReg' or 'contrastReg'. Allowed values include one of 'randomForest', 'boosting' and 'logistic' (fastest). Default is 'randomForest'.

ipcw.model A formula describing the inverse probability of censoring weighting (IPCW) model to be fitted. The left-hand side must be empty. Default is ipcw.model = NULL, which corresponds to specifying the IPCW model with the same covariates as the outcome model cate.model plus the treatment.

ipcw.method A character value for the censoring model. Allowed values are: 'breslow' (Cox regression with Breslow estimator of the baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)' (accelerated failure time model with different distributions for y variable). Default is 'breslow'.

minPS A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.

maxPS A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

followup.time A column name in data specifying the maximum follow-up time, interpreted as the potential censoring time. Default is followup.time = NULL, which corresponds to unknown potential censoring time.

tau0 The truncation time for defining restricted mean time lost. Default is NULL, which corresponds to setting the truncation time as the maximum survival time in the data.

higher.y A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.

prop.cutoff A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).

surv.min — Lower truncation limit for the probability of being censored. It must be a positive value and should be chosen close to 0. Default is 0.025.

tree.depth — A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 2.

n.trees.rf — A positive integer specifying the maximum number of trees in random forest. Used if score.method = 'ranfomForest' or if initial.predictor.method = 'randomForest' with score.method = 'twoReg' or 'contrastReg'. Only applies for survival outcomes. Default is 1000.

n.trees.boosting — A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 200.

B — A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold — A positive integer specifying the number of folds used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 5.

plot.gbmperf — A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

error.maxNR — A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

max.iterNR — A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.

tune — A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

seed — An optional integer specifying an initial randomization seed for reproducibility. Default is NULL, corresponding to no seed.

verbose — An integer value indicating what kind of intermediate progress messages should be printed. 0 means no outputs. 1 means only progress and run time. 2 means progress, run time, and all errors and warnings. Default is 0.

... — Additional arguments for gbm()

Details

The CATE score represents an individual-level treatment effect for survival data, estimated with random forest, boosting, Poisson regression, and the doubly robust estimator (two regressions, Yadlowsky, 2020) applied separately by treatment group or with the other doubly robust estimators (contrast regression, Yadlowsky, 2020) applied to the entire data set.

catefitsurv() provides the coefficients of the CATE score for each scoring method requested through score.method. Currently, contrast regression is the only method which allows for inference of the CATE coefficients by providing standard errors of the coefficients. The coefficients can be used to learn the effect size of each variable and predict the CATE score for a new observation.

catefitsurv() also provides the predicted CATE score of each observation in the data set, for each scoring method. The predictions allow ranking the observations from potentially high responders to the treatment to potentially low or standard responders.

The estimated ATE among nested subgroups of high responders are also provided by scoring method. Note that the ATEs in catefitsurv() are derived based on the CATE score which is estimated using the same data sample. Therefore, overfitting may be an issue. catecvsurv() is more suitable to inspect the estimated ATEs across scoring methods as it implements internal cross validation to reduce optimism.

Value

Returns an object of the class catefit containing the following components:

- ate.randomForest: A vector of numerical values of length prop.cutoff containing the estimated ATE by the RMTL ratio in nested subgroups (defined by prop.cutoff) constructed based on the estimated CATE scores with random forest method. Only provided if score.method includes 'randomForest'.
- ate.boosting: Same as ate.randomForest, but with the nested subgroups based the estimated CATE scores with boosting. Only provided if score.method includes 'boosting'.
- ate.poisson: Same as ate.randomForest, but with the nested subgroups based the estimated CATE scores with poisson regression. Only provided if score.method includes 'poisson'.
- ate.twoReg: Same as ate.randomForest, but with the nested subgroups based the estimated CATE scores with two regressions. Only provided if score.method includes 'twoReg'.
- ate.contrastReg: Same as ate.randomForest, but with the nested subgroups based the estimated CATE scores with contrast regression. Only provided if score.method includes 'contrastReg'.
- hr.randomForest: A vector of numerical values of length prop.cutoff containing the adjusted hazard ratio in nested subgroups (defined by prop.cutoff) constructed based on the estimated CATE scores with random forest method. Only provided if score.method includes 'randomForest'.
- hr.boosting: Same as hr.randomForest, but with the nested subgroups based the estimated CATE scores with boosting. Only provided if score.method includes 'boosting'.
- hr.poisson: Same as hr.randomForest, but with the nested subgroups based the estimated CATE scores with poisson regression. Only provided if score.method includes 'poisson'.
- hr.twoReg: Same as hr.randomForest, but with the nested subgroups based the estimated CATE scores with two regressions. Only provided if score.method includes 'twoReg'.
- hr.contrastReg: Same as hr.randomForest, but with the nested subgroups based the estimated CATE scores with contrast regression. Only provided if score.method includes 'contrastReg'.

- `score.randomForest`: A vector of numerical values of length n (number of observations in `data`) containing the estimated log-CATE scores according to random forest. Only provided if `score.method` includes `'randomForest'`.
- `score.boosting`: Same as `score.randomForest`, but with estimated log-CATE score according to boosting. Only provided if `score.method` includes `'boosting'`.
- `score.poisson`: Same as `score.randomForest`, but with estimated log-CATE score according to the Poisson regression. Only provided if `score.method` includes `'poisson'`.
- `score.twoReg`: Same as `score.randomForest`, but with estimated log-CATE score according to two regressions. Only provided if `score.method` includes `'twoReg'`.
- `score.contrastReg`: Same as `score.randomForest`, but with estimated log-CATE score according to contrast regression. Only provided if `score.method` includes `'contrastReg'`.
- `fit`: Additional details on model fitting if `score.method` includes 'randomForest', 'boosting' or 'contrastReg':
  - `result.randomForest`: Details on the random forest model fitted to observations with treatment = 0 (`$fit0.rf`) and to observations with treatment = 1 (`$fit1.rf`). Only provided if `score.method` includes `'randomForest'`.
  - `result.boosting`: Details on the boosting model fitted to observations with treatment = 0, (`$fit0.boosting`) and (`$fit0.gam`) and to observations with treatment = 1, (`$fit1.boosting`) and (`$fit1.gam`). Only provided if `score.method` includes `'boosting'`.
  - `result.contrastReg$converge.contrastReg`: Whether the contrast regression algorithm converged or not. Only provided if `score.method` includes `'contrastReg'`.
- `coefficients`: A data frame with the coefficients of the estimated log-CATE score by `score.method`. The data frame has number of rows equal to the number of covariates in `cate.model` and number of columns equal to `length(score.method)`. If `score.method` includes `'contrastReg'`, the data frame has an additional column containing the standard errors of the coefficients estimated with contrast regression. `'randomForest'` and `'boosting'` do not have coefficient results because tree-based methods typically do not express the log-CATE as a linear combination of coefficients and covariates.
- `errors/warnings`: A nested list of errors and warnings that were wrapped during the calculation of ATE. Errors and warnings are organized by `score.method`.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

`catecvsurv()`

Examples

```
library(survival)

tau0 <- with(survivalExample, min(quantile(y[trt == "drug1"], 0.95),
                                  quantile(y[trt == "drug0"], 0.95)))
```

```
fit <- catefitsurv(data = survivalExample,
                   score.method = "randomForest",
                   cate.model = Surv(y, d) ~ age + female + previous_cost +
                                             previous_number_relapses,
                   ps.model = trt ~ age + previous_treatment,
                   ipcw.model = ~ age + previous_cost + previous_treatment,
                   tau0 = tau0,
                   seed = 999)

coef(fit)
```

`countExample` *Simulated data with count outcome*

Description

A dataset containing a count outcome, a length of follow-up and 6 baseline covariates

Usage

```
data(countExample)
```

Format

A dataframe with 4000 rows (patients) and 9 variables:

age age at baseline, centered to 48 years old, in years

female sex, 0 for male, 1 for female

previous_treatment previous treatment, "drugA", "drugB", or "drugC"

previous_cost previous medical cost, in US dollars

previous_number_symptoms previous number of symptoms, "0", "1", or ">=2"

previous_number_relapses previous number of relapses

trt current treatment, "drug0" or "drug1"

y count outcome, current number of relapses

years length of follow-up, in years

Examples

```
data(countExample)
str(countExample)
rate <- countExample$y / countExample$years
```

cox.rmst — *Estimate restricted mean survival time (RMST) based on Cox regression model*

Description

Estimate restricted mean survival time (RMST) based on Cox regression model

Usage

```
cox.rmst(y, d, x.cate, xnew, tau0)
```

Arguments

y	Observed survival or censoring time; vector of size n.
d	The event indicator, normally 1 = event, 0 = censored; vector of size n.
x.cate	Matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate.
xnew	Matrix of p.cate baseline covariates for which we want an estimate of the RMST; dimension m (observations in the new data set) by p.cate
tau0	The truncation time for defining restricted mean time lost.

Value

The estimated RMST for new subjects with covariates xnew; vector of size m.

data.preproc — *Data preprocessing Apply at the beginning of* pmcount() *and* cvcount()*, after* arg.checks()

Description

Data preprocessing Apply at the beginning of pmcount() and cvcount(), after arg.checks()

Usage

```
data.preproc(
  fun,
  cate.model,
  ps.model,
  data,
  prop.cutoff = NULL,
  prop.multi = NULL,
  ps.method,
  initial.predictor.method = NULL
)
```

Arguments

fun	A function for which argument check is needed; "pm" for pmcount(), "cv" for cvcount(), and "drinf" for drcount.inference(). No default.
cate.model	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
ps.model	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a RCT, specify ps.model as an intercept-only model.
data	A data frame containing the variables in the outcome and propensity score models; a data frame with n rows (1 row per observation).
prop.cutoff	A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of prop.cutoff is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is seq(0.5, 1, length = 6).
prop.multi	A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of length(prop.multi) > 2. Each element represents the cutoff to separate the observations into length(prop.multi) - 1 mutually exclusive subgroups. Default is c(0, 1/3, 2/3, 1).
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
initial.predictor.method	A character vector for the method used to get initial outcome predictions conditional on the covariates. Only applies when score.method includes 'twoReg' or 'contrastReg'. Allowed values include one of 'boosting', 'poisson' (fast), and 'gam'. Default is NULL, which assigns 'boosting' for count outcomes.

Value

A list of 6 elements: - y: outcome; vector of length n (observations) - trt: binary treatment; vector of length n - x.ps: matrix of p.ps baseline covariates (plus intercept); dimension n by p.ps + 1 - x.cate: matrix of p.cate baseline covariates; dimension n by p.cate - time: offset; vector of length n - if fun = "pm": - prop: formatted prop.cutoff - if fun = "cv" - prop.onlyhigh: formatted prop.cutoff with 0 removed if applicable - prop.bi; formatted prop.cutoff with 0 and 1 removed if applicable - prop.multi: formatted prop.multi, starting with 0 and ending with 1

`data.preproc.mean` *Data preprocessing Apply at the beginning of* `catefitmean()` *and* `catecvmean()`*, after* `arg.checks()`

Description

Data preprocessing Apply at the beginning of `catefitmean()` and `catecvmean()`, after `arg.checks()`

Usage

```
data.preproc.mean(
  fun,
  cate.model,
  init.model,
  ps.model,
  data,
  prop.cutoff = NULL,
  prop.multi = NULL,
  ps.method,
  score.method = NULL,
  initial.predictor.method = NULL
)
```

Arguments

`fun`	A function for which argument check is needed; "pm" for `catefitmean()`, "cv" for `catecvmean()`, and "drinf" for `drmean.inference()`. No default.
`cate.model`	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
`init.model`	A formula describing the initial predictor model. The outcome must appear on the left-hand side. It must be specified when `score.method` = `contrastReg` or `twoReg`.
`ps.model`	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a RCT, specify `ps.model` as an intercept-only model.
`data`	A data frame containing the variables in the outcome and propensity score models; a data frame with `n` rows (1 row per observation).
`prop.cutoff`	A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cutoff to separate observations in nested subgroups (below vs above cutoff). The length of `prop.cutoff` is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is `seq(0.5, 1, length = 6)`.

prop.multi A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of length(prop.multi) > 2. Each element represents the cutoff to separate the observations into length(prop.multi) - 1 mutually exclusive subgroups. Default is c(0, 1/3, 2/3, 1).

ps.method A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.

score.method A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'boosting', 'gaussian', 'twoReg', 'contrastReg', 'randomForest', 'gam'.

initial.predictor.method
A character vector for the method used to get initial outcome predictions conditional on the covariates. Only applies when score.method includes 'twoReg' or 'contrastReg'. Allowed values include one of 'boosting', 'poisson' (fast), and 'gam'. Default is NULL, which assigns 'boosting' for count outcomes.

Value

A list of 6 elements: - y: outcome; vector of length n (observations) - trt: binary treatment; vector of length n - x.ps: matrix of p.ps baseline covariates (plus intercept); dimension n by p.ps + 1 - x.cate: matrix of p.cate baseline covariates; dimension n by p.cate - x.init: matrix of p.init baseline covariates; dimension n by p.init - if fun = "pm": - prop: formatted prop.cutoff - if fun = "cv" - prop.onlyhigh: formatted prop.cutoff with 0 removed if applicable - prop.bi; formatted prop.cutoff with 0 and 1 removed if applicable - prop.multi: formatted prop.multi, starting with 0 and ending with 1

---

data.preproc.surv *Data preprocessing Apply at the beginning of* catefitcount(), catecvcount(), catefitsurv(), *and* catecvsurv(), *after* arg.checks()

Description

Data preprocessing Apply at the beginning of catefitcount(), catecvcount(), catefitsurv(), and catecvsurv(), after arg.checks()

Usage

```
data.preproc.surv(
  fun,
  cate.model,
  ps.model,
```

```
  ipcw.model = NULL,
  tau0 = NULL,
  data,
  prop.cutoff = NULL,
  prop.multi = NULL,
  ps.method,
  initial.predictor.method = NULL,
  response = "count"
)
```

Arguments

fun	A function for which argument check is needed; "catefit" for `catefitcount()` and `catefitsurv()`, "crossv" for `catecvcount()` and `catecvsurv()`, and "drinf" for `drcount.inference()` and `drsurv.inference()`. No default.
cate.model	A formula describing the outcome model to be fitted. The outcome must appear on the left-hand side.
ps.model	A formula describing the propensity score model to be fitted. The treatment must appear on the left-hand side. The treatment must be a numeric vector coded as 0/1. If data are from a RCT, specify `ps.model` as an intercept-only model.
ipcw.model	A formula describing inverse probability of censoring weighting(IPCW) model to be fitted. If covariates are the same as outcome model, set `ipcw.model = NULL`. Otherwise, the left-hand side must be empty and the right-hand side is a covariates model.
tau0	The truncation time for defining restricted mean time lost. Default is `NULL`, which corresponds to setting the truncation time as the maximum survival time in the data
data	A data frame containing the variables in the outcome, propensity score, and IPCW models; a data frame with `n` rows (1 row per observation).
prop.cutoff	A vector of numerical values (in '(0, 1]') specifying percentiles of the estimated log CATE scores to define nested subgroups. Each element represents the cut-off to separate observations in nested subgroups (below vs above cutoff). The length of `prop.cutoff` is the number of nested subgroups. An equally-spaced sequence of proportions ending with 1 is recommended. Default is `seq(0.5, 1, length = 6)`.
prop.multi	A vector of numerical values (in '[0, 1]') specifying percentiles of the estimated log CATE scores to define mutually exclusive subgroups. It should start with 0, end with 1, and be of `length(prop.multi) > 2`. Each element represents the cutoff to separate the observations into `length(prop.multi) - 1` mutually exclusive subgroups. Default is `c(0, 1/3, 2/3, 1)`.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: `'glm'` for logistic regression with main effects only (default), or `'lasso'` for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in `ps.model`). Relevant only when `ps.model` has more than one variable.

initial.predictor.method
: A character vector for the method used to get initial outcome predictions conditional on the covariates. Only applies when score.method includes 'twoReg' or 'contrastReg'. Allowed values include one of 'randomForest' (survival outcomes only), 'boosting', 'logistic' (survival outcomes only, fast), 'poisson' (count outcomes only, fast), and 'gam' (count outcomes only). Default is NULL, which assigns 'boosting' for count outcomes and 'randomForest' for survival outcomes.

response
: The type of response variables; count (default) or survival.

Value

A list of elements: - y: outcome; vector of length n (observations) - d : the event indicator; vector of length n; only if respone = "survival" - trt: binary treatment; vector of length n - x.ps: matrix of p.ps baseline covariates specified in the propensity score model (plus intercept); dimension n by p.ps + 1 - x.cate: matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate - x.ipcw: matrix of p.ipw baseline covarites specified in inverse probability of censoring weighting model; dimension n by p.ipw - time: offset; vector of length n; only if response = "count" - if fun = "catefit": - prop: formatted prop.cutoff - prop.no1: formatted prop.cutoff with 1 removed if applicable; otherwise prop.no1 is the same as prop - if fun = "crossv" - prop.onlyhigh: formatted prop.cutoff with 0 removed if applicable - prop.bi; formatted prop.cutoff with 0 and 1 removed if applicable - prop.multi: formatted prop.multi, starting with 0 and ending with 1

---

drcount *Doubly robust estimator of the average treatment effect for count data*

Description

Doubly robust estimator of the average treatment effect between two treatments, which is the rate ratio of treatment 1 over treatment 0 for count outcomes.

Usage

```
drcount(
  y,
  trt,
  x.cate,
  x.ps,
  time,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  interactions = TRUE
)
```

Arguments

y	A numeric vector of size n with each element representing the observed count outcome for each subject.
trt	A numeric vector (in {0, 1}) of size n with each element representing the treatment received for each subject.
x.cate	A numeric matrix of dimension n by p.cate with each column representing each baseline covariate specified in the outcome model for all subjects.
x.ps	A numeric matrix of dimension n by p.ps + 1 with a leading column of 1 as the intercept and each remaining column representing each baseline covariate specified in the propensity score model for all subjects.
time	A numeric vector of size n with each element representing the log-transformed person-years of follow-up for each subject.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value between 0 and 1 below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value between 0 and 1 above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
interactions	A logical value indicating whether the outcome model should allow for treatment-covariate interaction by x. If TRUE, interactions will be assumed only if at least 10 patients received each treatment option. Default is TRUE.

Value

Return a list of 4 elements:

- log.rate.ratio: A numeric value of the estimated log rate ratio.
- rate0: A numeric value of the estimated rate in the group trt=0.
- rate1: A numeric value of the estimated rate in the group trt=1.

---

drmean *Doubly robust estimator of the average treatment effect for continuous data*

Description

Doubly robust estimator of the average treatment effect between two treatments, which is the mean difference of treatment 1 over treatment 0 for continuous outcomes.

Usage

```
drmean(
  y,
  trt,
  x.cate,
  x.ps,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  interactions = TRUE
)
```

Arguments

y	A numeric vector of size n with each element representing the observed continuous outcome for each subject.
trt	A numeric vector (in {0, 1}) of size n with each element representing the treatment received for each subject.
x.cate	A numeric matrix of dimension n by p.cate with each column representing each baseline covariate specified in the outcome model for all subjects.
x.ps	A numeric matrix of dimension n by p.ps + 1 with a leading column of 1 as the intercept and each remaining column representing each baseline covariate specified in the propensity score model for all subjects
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value between 0 and 1 below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value between 0 and 1 above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
interactions	A logical value indicating whether the outcome model should assume interactions between x and trt. If TRUE, interactions will be assumed only if at least 10 patients received each treatment option. Default is TRUE.

Value

Return a list of 4 elements:

- mean.diff: A numeric value of the estimated mean difference.
- mean.diff0: A numeric value of the estimated mean difference in treatment group 0.
- mean.diff1: A numeric value of the estimated mean difference in treatment group 1.

drsurv *Doubly robust estimator of the average treatment effect with Cox model for survival data*

Description

Doubly robust estimator of the average treatment effect between two treatments, which is the restricted mean time lost (RMTL) ratio of treatment 1 over treatment 0 for survival outcomes.

Usage

```
drsurv(
  y,
  d,
  x.cate,
  x.ps,
  x.ipcw,
  trt,
  yf = NULL,
  tau0,
  surv.min = 0.025,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  ipcw.method = "breslow"
)
```

Arguments

y	Observed survival or censoring time; vector of size n.
d	The event indicator, normally 1 = event, 0 = censored; vector of size n.
x.cate	Matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate.
x.ps	Matrix of p.ps baseline covariates specified in the propensity score model; dimension n by p.ps.
x.ipcw	Matrix of p.ipw baseline covariate specified in inverse probability of censoring weighting; dimension n by p.ipw.
trt	Treatment received; vector of size n with treatment coded as 0/1.
yf	Follow-up time, interpreted as the potential censoring time; vector of size n if the potential censoring time is known.
tau0	The truncation time for defining restricted mean time lost.
surv.min	Lower truncation limit for probability of being censored (positive and very close to 0).

ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
ipcw.method	The censoring model. Allowed values are: 'breslow' (Cox regression with Breslow estimator of the baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)'. Default is 'breslow'.

Value

Return a list of 4 elements:

- rmst1: A numeric value of the estimated restricted mean survival time n the group trt = 1.
- rmst0: A numeric value of the estimated restricted mean survival time n the group trt = 0.
- log.rmtl.ratio: A numeric value of the estimated log rmtl ratio.
- log.hazard.ratio: A numeric value of the estimated log hazard ratio.

---

estcount.bilevel.subgroups

Estimate the Average Treatment Effect of the log risk ratio in multiple bi-level subgroups defined by the proportions

---

Description

If only care about the higher subgroup (above cutoff), only need trt.est.high so set onlyhigh to be TRUE Scores are adjusted to the opposite sign if higher.y == FALSE; scores stay the same if higher.y == TRUE; this is because estcount.bilevel.subgroups() always takes the subgroup of the top highest adjusted scores, and higher adjusted scores should always represent high responders of trt=1

Usage

```
estcount.bilevel.subgroups(
  y,
  x.cate,
  x.ps,
  time,
  trt,
  score,
  higher.y,
```

```
  prop,
  onlyhigh,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99
)
```

Arguments

y	Observed outcome; vector of size n (observations)
x.cate	Matrix of p.cate baseline covariates; dimension n by p.cate (covariates in the outcome model)
x.ps	Matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept); dimension n by p.ps + 1 (covariates in the propensity score model plus intercept)
time	Log-transformed person-years of follow-up; vector of size n
trt	Treatment received; vector of size n units with treatment coded as 0/1
score	Estimated log CATE scores for all n observations from one of the four methods (boosting, naive Poisson, two regressions, contrast regression); vector of size n
higher.y	A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.
prop	Proportions corresponding to percentiles in the estimated log CATE scores that define subgroups to calculate ATE for; vector of floats in '(0, 1]' (if onlyhigh=T) or in '(0, 1)' (if onlyhigh=F): Each element of prop represents the high/low cutoff in each bi-level subgroup and the length of prop is number of bi-level subgroups
onlyhigh	Indicator of returning only the ATEs in the higher-than-cutoff category of the bi-level subgroups; boolean
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

Value

ate.est.high: estimated ATEs in the multiple bi-level subgroups that are in the higher-than-cutoff category; vector of size equal to the length of prop; always returned ate.est.low: estimated ATEs in the multiple bi-level subgroups that are in the lower-than-cutoff category; vector of size equal to the length of prop; returned only when onlyhigh == TRUE

`estcount.multilevel.subgroup` *Estimate the ATE of the log RR ratio in one multilevel subgroup defined by the proportions*

Description

Scores are adjusted to the opposite sign if `higher.y` == FALSE; scores stay the same if `higher.y` == TRUE; this is because subgroups defined in estcount.multilevel.subgroup() start from the lowest to the highest adjusted scores, and higher adjusted scores should always represent high responders of trt=1

Usage

```
estcount.multilevel.subgroup(
  y,
  x.cate,
  x.ps,
  time,
  trt,
  score,
  higher.y,
  prop,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99
)
```

Arguments

`y`	Observed outcome; vector of size `n` (observations)
`x.cate`	Matrix of `p.cate` baseline covariates; dimension `n` by `p.cate` (covariates in the outcome model)
`x.ps`	Matrix of `p.ps` baseline covariates (plus a leading column of 1 for the intercept); dimension `n` by `p.ps` + 1 (covariates in the propensity score model plus intercept)
`time`	Log-transformed person-years of follow-up; vector of size `n`
`trt`	Treatment received; vector of size `n` units with treatment coded as 0/1
`score`	Estimated log CATE scores for all `n` observations from one of the four methods (boosting, naive Poisson, two regressions, contrast regression); vector of size `n`
`higher.y`	A logical value indicating whether higher (`TRUE`) or lower (`FALSE`) values of the outcome are more desirable. Default is `TRUE`.
`prop`	Proportions corresponding to percentiles in the estimated log CATE scores that define subgroups to calculate ATE for; vector of floats in '[0, 1]' always starting with 0 and ending with 1: Each element of `prop` represents inclusive cutoffs in

	the multilevel subgroup and the length of prop is number of categories in the multilevel subgroup
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

Value

estimated ATEs of all categories in the one multilevel subgroup; vector of size equal to the length of categories in the multilevel subgroup

---

estmean.bilevel.subgroups

Estimate the ATE of the mean difference in multiple bi-level subgroups defined by the proportions

---

Description

If only care about the higher subgroup (above cutoff), only need trt.est.high so set onlyhigh to be TRUE. Scores are adjusted to the opposite sign if higher.y == FALSE; scores stay the same if higher.y == TRUE. This is because estcount.bilevel.subgroups() always takes the subgroup of the top highest adjusted scores,and higher adjusted scores should always represent high responders in treatment group 1.

Usage

```
estmean.bilevel.subgroups(
  y,
  x.cate,
  x.ps,
  trt,
  score,
  higher.y,
  prop,
  onlyhigh,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99
)
```

Arguments

y	Observed outcome; vector of size n (observations)
x.cate	Matrix of p.cate baseline covariates; dimension n by p.cate (covariates in the outcome model)
x.ps	Matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept); dimension n by p.ps + 1 (covariates in the propensity score model plus intercept)
trt	Treatment received; vector of size n units with treatment coded as 0/1
score	Estimated CATE scores for all n observations from one of the six methods (boosting, linear regression, two regressions, contrast regression, random forest, generalized additive model); vector of size n
higher.y	A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.
prop	Proportions corresponding to percentiles in the estimated CATE scores that define subgroups to calculate ATE for; vector of floats in '(0, 1]' (if onlyhigh=T) or in '(0, 1)' (if onlyhigh=F): Each element of prop represents the high/low cutoff in each bi-level subgroup and the length of prop is number of bi-level subgroups
onlyhigh	Indicator of returning only the ATEs in the higher-than-cutoff category of the bi-level subgroups; boolean
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

Value

ate.est.high: estimated ATEs in the multiple bi-level subgroups that are in the higher-than-cutoff category; vector of size equal to the length of prop; always returned ate.est.low: estimated ATEs in the multiple bi-level subgroups that are in the lower-than-cutoff category; vector of size equal to the length of prop; returned only when onlyhigh == TRUE

---

estmean.multilevel.subgroup

Estimate the ATE of the mean difference in one multilevel subgroup defined by the proportions

---

Description

Scores are adjusted to the opposite sign if higher.y == FALSE; scores stay the same if higher.y == TRUE; this is because subgroups defined in estmean.multilevel.subgroup() start from the lowest to the highest adjusted scores, and higher adjusted scores should always represent high responders of trt=1

Usage

```
estmean.multilevel.subgroup(
  y,
  x.cate,
  x.ps,
  trt,
  score,
  higher.y,
  prop,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99
)
```

Arguments

y	Observed outcome; vector of size n (observations)
x.cate	Matrix of p.cate baseline covariates; dimension n by p.cate (covariates in the outcome model)
x.ps	Matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept); dimension n by p.ps + 1 (covariates in the propensity score model plus intercept)
trt	Treatment received; vector of size n units with treatment coded as 0/1
score	Estimated CATE scores for all n observations from one of the six methods (boosting, linear regression, two regressions, contrast regression, random forest, generalized additive model); vector of size n
higher.y	A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.
prop	Proportions corresponding to percentiles in the estimated CATE scores that define subgroups to calculate ATE for; vector of floats in '[0, 1]' always starting with 0 and ending with 1: Each element of prop represents inclusive cutoffs in the multilevel subgroup and the length of prop is number of categories in the multilevel subgroup
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.

`minPS` A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is `0.01`.

`maxPS` A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than `minPS`. Default is `0.99`.

Value

estimated ATEs of all categories in the one multilevel subgroup; vector of size equal to the length of categories in the multilevel subgroup

---

`estsurv.bilevel.subgroups` *Estimate the ATE of the RMTL ratio and unadjusted hazard ratio in multiple bi-level subgroups defined by the proportions*

Description

If only care about the higher subgroup (above cutoff), only need ate.rmtl.high and hr.high so set "onlyhigh" to be TRUE Scores are adjusted to the opposite sign if `higher.y` == FALSE; scores stay the same if `higher.y` == FALSE; this is because estsurv() function always takes the subgroup of the top highest adjusted scores, and higher adjusted scores should always represent high responders of trt=1

Usage

```
estsurv.bilevel.subgroups(
  y,
  d,
  x.cate,
  x.ps,
  x.ipcw,
  trt,
  yf,
  tau0 = tau0,
  score,
  higher.y,
  prop,
  onlyhigh,
  surv.min = 0.025,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  ipcw.method = "breslow"
)
```

Arguments

y	Observed survival or censoring time; vector of size n.
d	The event indicator, normally 1 = event, 0 = censored; vector of size n.
x.cate	Matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate.
x.ps	Matrix of p.ps baseline covariates specified in the propensity score model; dimension n by p.ps.
x.ipcw	Matrix of p.ipw baseline covariate specified in inverse probability of censoring weighting; dimension n by p.ipw.
trt	Treatment received; vector of size n with treatment coded as 0/1.
yf	Follow-up time, interpreted as the potential censoring time; vector of size n if the potential censoring time is known.
tau0	The truncation time for defining restricted mean time lost.
score	Estimated log CATE scores for all n observations from one of the five methods (random forest, boosting, naive Poisson, two regressions, contrast regression); vector of size n.
higher.y	A logical value indicating whether higher (TRUE) or lower (FALSE)
prop	Proportions corresponding to percentiles in the estimated log CATE scores that define subgroups to calculate ATE for; vector of floats in '(0, 1]' (if onlyhigh=TRUE) or in '(0, 1)' (if onlyhigh=FALSE): Each element of prop represents the high/low cutoff in each bi-level subgroup and the length of prop is number of bi-level subgroups
onlyhigh	Indicator of returning only the ATEs in the higher-than-cutoff category of the bi-level subgroups; boolean.
surv.min	Lower truncation limit for probability of being censored (positive and very close to 0).
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
ipcw.method	The censoring model. Allowed values are: 'breslow' (Cox regression with Breslow estimator of the baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)'. Default is 'breslow'.

Value

ate.rmtl.high: estimated ATEs (ratio of RMTL) in the multiple bi-level subgroups that are in the higher-than-cutoff category; vector of size equal to the length of prop; always returned. ate.rmtl.low: estimated ATEs (ratio of RMTL) in the multiple bi-level subgroups that are in the lower-than-cutoff category; vector of size equal to the length of prop; returned only when `onlyhigh = TRUE`. hr.high: unadjusted hazard ratio in the multiple bi-level subgroups that are in the higher-than-cutoff category; vector of size equal to the length of prop; always returned. hr.low: unadjusted hazard ratio in the multiple bi-level subgroups that are in the lower-than-cutoff category; vector of size equal to the length of prop; returned only when `onlyhigh = TRUE`

`estsurv.multilevel.subgroups`

Estimate the ATE of the RMTL ratio and unadjusted hazard ratio in one multilevel subgroup defined by the proportions

---

Description

Scores are adjusted to the opposite sign if `higher.y` == FALSE; scores stay the same if `higher.y` == FALSE; this is because estsurv function for multilevel subgroups start from the lowest to the highest adjusted scores, and higher adjusted scores should always represent high responders of trt=1

Usage

```
estsurv.multilevel.subgroups(
  y,
  d,
  x.cate,
  x.ps,
  x.ipcw,
  trt,
  yf,
  tau0 = tau0,
  score,
  higher.y,
  prop,
  surv.min = 0.025,
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  ipcw.method = "breslow"
)
```

Arguments

`y`	Observed survival or censoring time; vector of size `n`.

d	The event indicator, normally 1 = event, 0 = censored; vector of size n.
x.cate	Matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate.
x.ps	Matrix of p.ps baseline covariates specified in the propensity score model; dimension n by p.ps.
x.ipcw	Matrix of p.ipw baseline covariate specified in inverse probability of censoring weighting; dimension n by p.ipw.
trt	Treatment received; vector of size n with treatment coded as 0/1.
yf	Follow-up time, interpreted as the potential censoring time; vector of size n if the potential censoring time is known.
tau0	The truncation time for defining restricted mean time lost.
score	Estimated log CATE scores for all n observations from one of the five methods (random forest, boosting, naive Poisson, two regressions, contrast regression); vector of size n.
higher.y	A logical value indicating whether higher (TRUE) or lower (FALSE) values of the outcome are more desirable. Default is TRUE.
prop	Proportions corresponding to percentiles in the estimated log CATE scores that define subgroups to calculate ATE for; vector of floats in '[0, 1]' always starting with 0 and ending with 1: Each element of prop represents inclusive cutoffs in the multilevel subgroup and the length of prop is number of categories in the multilevel subgroup
surv.min	Lower truncation limit for probability of being censored (positive and very close to 0).
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.
ipcw.method	The censoring model. Allowed values are: 'breslow' (Cox regression with Breslow estimator of the baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)'. Default is 'breslow'.

Value

ate.rmtl: estimated ATEs (ratio of RMTL) of all categories in the one multilevel subgroup; vector of size equal to the length of categories in the multilevel subgroup. ate.hr: unadjusted hazard ratio of all categories in the one multilevel subgroup; vector of size equal to the length of categories in the multilevel subgroup.

generate_kfold_indices *Generate K-fold Indices for Cross-Validation*

Description

This function generates indices for K-fold cross-validation based on the total sample size 'N' and the number of folds 'Kfold'. If 'reverse = TRUE', the remainder indices will be assigned in reverse order.

Usage

```
generate_kfold_indices(N, Kfold, reverse = FALSE)
```

Arguments

N	Integer. Total sample size (number of observations).
Kfold	Integer. The number of folds to split the data into.
reverse	Logical. Whether to reverse the remainder indices when 'N' is not divisible by 'Kfold'. Defaults to 'FALSE'.

Value

A vector of length 'N' containing the fold assignments (from 1 to 'Kfold').

Author(s)

Thomas Debray

glm.ps *Propensity score estimation with LASSO*

Description

Propensity score based on a multivariate logistic regression with LASSO penalization on the two-way interactions

Usage

```
glm.ps(trt, x.ps, xnew = NULL, minPS = 0.01, maxPS = 0.99)
```

Arguments

trt	Treatment received; vector of size n (observations) with treatment coded as 0/1
x.ps	Matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept); dimension n by p.ps + 1 (covariates in the propensity score model plus intercept)
xnew	Matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept) for which we want propensity scores predictions; dimension m (observations in the new data set) by p.ps + 1
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

Value

The trimmed propensity score for each unit; vector of size n (if xnew is NULL) or m

---

glm.simplereg.ps *Propensity score estimation with a linear model*

Description

Propensity score based on a multivariate logistic regression with main effects only

Usage

```
glm.simplereg.ps(trt, x.ps, xnew = NULL, minPS = 0.01, maxPS = 0.99)
```

Arguments

trt	Treatment received; vector of size n (observations) with treatment coded as 0/1
x.ps	A matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept); dimension n by p.ps + 1 (covariates in the propensity score model plus intercept)
xnew	A matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept) for which we want PS predictions; dimension m (observations in the new data set) by p.ps + 1
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A numerical value (in '(0, 1]') above which estimated propensity scores should be truncated. Must be strictly greater than minPS. Default is 0.99.

Value

The estimated propensity score for each unit; vector of size n (if xnew is NULL) or m

`intxcount` *Estimate the CATE model using specified scoring methods*

Description

Coefficients of the CATE estimated with boosting, naive Poisson, two regression, contrast regression, negative binomial

Usage

```
intxcount(
  y,
  trt,
  x.cate,
  x.ps,
  time,
  score.method = c("boosting", "poisson", "twoReg", "contrastReg", "negBin"),
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  initial.predictor.method = "boosting",
  xvar.smooth = NULL,
  tree.depth = 2,
  n.trees.boosting = 200,
  B = 3,
  Kfold = 6,
  plot.gbmperf = TRUE,
  error.maxNR = 0.001,
  max.iterNR = 150,
  tune = c(0.5, 2),
  ...
)
```

Arguments

`y`	Observed outcome; vector of size `n` (observations)
`trt`	Treatment received; vector of size `n` units with treatment coded as 0/1
`x.cate`	Matrix of `p.cate` baseline covariates; dimension `n` by `p.cate` (covariates in the outcome model)
`x.ps`	Matrix of `p.ps` baseline covariates (plus a leading column of 1 for the intercept); dimension `n` by `p.ps + 1` (covariates in the propensity score model plus intercept)
`time`	Log-transformed person-years of follow-up; vector of size `n`
`score.method`	A vector of one or multiple methods to estimate the CATE score. Allowed values are: `'boosting'`, `'poisson'`, `'twoReg'`, `'contrastReg'`, `'negBin'`. Default specifies all 5 methods.

ps.method A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.

minPS A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.

maxPS A number above which estimated propensity scores should be trimmed; scalar

initial.predictor.method
A character vector for the method used to get initial outcome predictions conditional on the covariates in cate.model in score.method = 'twoReg' and 'contrastReg'. Allowed values include one of 'poisson' (fastest), 'boosting' (default) and 'gam'.

xvar.smooth A vector of characters indicating the name of the variables used as the smooth terms if initial.predictor.method = 'gam'. The variables must be selected from the variables listed in cate.model. Default is NULL, which uses all variables in cate.model.

tree.depth A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 2.

n.trees.boosting
A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 200.

B A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold A positive integer specifying the number of folds (parts) used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 6.

plot.gbmperf A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

error.maxNR A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

max.iterNR A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.

tune A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

... Additional arguments for gbm()

Value

Depending on what score.method is, the outputs is a combination of the following: result.boosting: Results of boosting fit and best iteration, for trt = 0 and trt = 1 separately result.poisson: Naive Poisson estimator (beta1 - beta0); vector of length p.cate + 1 result.twoReg: Two regression estimator (beta1 - beta0); vector of length p.cate + 1 result.contrastReg: A list of the contrast regression results with 3 elements: $delta.contrastReg: Contrast regression DR estimator; vector of length p.cate + 1 $sigma.contrastReg: Variance covariance matrix for delta.contrastReg; matrix of size p.cate + 1 by p.cate + 1 $converge.contrastReg: Indicator that the Newton Raphson algorithm converged for delta_0; boolean result.negBin: Negative binomial estimator (beta1 - beta0); vector of length p.cate + 1 best.iter: Largest best iterations for boosting (if used) fgam: Formula applied in GAM (if used)

---

intxmean *Estimate the CATE model using specified scoring methods*

Description

Coefficients of the CATE estimated with boosting, linear regression, two regression, contrast regression, random forest, generalized additive model

Usage

```
intxmean(
  y,
  trt,
  x.cate,
  x.init,
  x.ps,
  score.method = c("boosting", "gaussian", "twoReg", "contrastReg", "gam",
    "randomForest"),
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  initial.predictor.method = "boosting",
  xvar.smooth.init,
  xvar.smooth.score,
  tree.depth = 2,
  n.trees.rf = 1000,
  n.trees.boosting = 200,
  B = 1,
  Kfold = 2,
  plot.gbmperf = TRUE,
  ...
)
```

Arguments

y	Observed outcome; vector of size n (observations)
trt	Treatment received; vector of size n units with treatment coded as 0/1
x.cate	Matrix of p.cate baseline covariates; dimension n by p.cate (covariates in the outcome model)
x.init	Matrix of p.init baseline covariates; dimension n by p.init It must be specified when score.method = contrastReg or twoReg.
x.ps	Matrix of p.ps baseline covariates (plus a leading column of 1 for the intercept); dimension n by p.ps + 1 (covariates in the propensity score model plus intercept)
score.method	A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'boosting', 'gaussian', 'twoReg', 'contrastReg', 'randomForest', 'gam'. Default specifies all 6 methods.
ps.method	A character value for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A number above which estimated propensity scores should be trimmed; scalar
initial.predictor.method	A character vector for the method used to get initial outcome predictions conditional on the covariates in cate.model in score.method = 'twoReg' and 'contrastReg'. Allowed values include one of 'gaussian' (fastest), 'boosting' (default) and 'gam'.
xvar.smooth.init	A vector of characters indicating the name of the variables used as the smooth terms if initial.predictor.method = 'gam'. The variables must be selected from the variables listed in init.model. Default is NULL, which uses all variables in init.model.
xvar.smooth.score	A vector of characters indicating the name of the variables used as the smooth terms if score.method = 'gam'. The variables must be selected from the variables listed in cate.model. Default is NULL, which uses all variables in cate.model.
tree.depth	A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 2.
n.trees.rf	A positive integer specifying the number of trees. Used only if score.method = 'randomForest'. Default is 1000.
n.trees.boosting	A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 200.

B	A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.
Kfold	A positive integer specifying the number of folds (parts) used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 6.
plot.gbmperf	A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.
...	Additional arguments for gbm()

Value

Depending on what score.method is, the outputs is a combination of the following: result.boosting: Results of boosting fit and best iteration, for trt = 0 and trt = 1 separately result.gaussian: Linear regression estimator (beta1 - beta0); vector of length p.cate + 1 result.twoReg: Two regression estimator (beta1 - beta0); vector of length p.cate + 1 result.contrastReg: A list of the contrast regression results with 3 elements: $delta.contrastReg: Contrast regression DR estimator; vector of length p.cate + 1 $sigma.contrastReg: Variance covariance matrix for delta.contrastReg; matrix of size p.cate + 1 by p.cate + 1 result.randomForest: Results of random forest fit and best iteration, for trt = 0 and trt = 1 separately result.gam: Results of generalized additive model fit and best iteration, for trt = 0 and trt = 1 separately best.iter: Largest best iterations for boosting (if used) fgam: Formula applied in GAM when initial.predictor.method = 'gam' warn.fit: Warnings occurred when fitting score.method err.fit:: Errors occurred when fitting score.method

---

intxsurv *Estimate the CATE model using specified scoring methods for survival outcomes*

Description

Coefficients of the CATE estimated with random forest, boosting, naive Poisson, two regression, and contrast regression

Usage

```
intxsurv(
  y,
  d,
  trt,
  x.cate,
  x.ps,
  x.ipcw,
  yf = NULL,
  tau0,
  surv.min = 0.025,
```

```
 score.method = c("randomForest", "boosting", "poisson", "twoReg", "contrastReg"),
  ps.method = "glm",
  minPS = 0.01,
  maxPS = 0.99,
  ipcw.method = "breslow",
  initial.predictor.method = "randomForest",
  tree.depth = 3,
  n.trees.rf = 1000,
  n.trees.boosting = 150,
  B = 3,
  Kfold = 5,
  plot.gbmperf = TRUE,
  error.maxNR = 0.001,
  max.iterNR = 100,
  tune = c(0.5, 2),
  ...
)
```

Arguments

y	Observed survival or censoring time; vector of size n.
d	The event indicator, normally 1 = event, 0 = censored; vector of size n.
trt	Treatment received; vector of size n with treatment coded as 0/1.
x.cate	Matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate.
x.ps	Matrix of p.ps baseline covariates specified in the propensity score model; dimension n by p.ps.
x.ipcw	Matrix of p.ipw baseline covariate specified in inverse probability of censoring weighting; dimension n by p.ipw.
yf	Follow-up time, interpreted as the potential censoring time; vector of size n if the potential censoring time is known.
tau0	The truncation time for defining restricted mean time lost.
surv.min	Lower truncation limit for probability of being censored (positive and very close to 0).
score.method	A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'randomForest', 'boosting', 'poisson', 'twoReg', 'contrastReg'. Default specifies all 5 methods.
ps.method	A character vector for the method to estimate the propensity score. Allowed values include one of: 'glm' for logistic regression with main effects only (default), or 'lasso' for a logistic regression with main effects and LASSO penalization on two-way interactions (added to the model if interactions are not specified in ps.model). Relevant only when ps.model has more than one variable.
minPS	A numerical value (in '[0, 1]') below which estimated propensity scores should be truncated. Default is 0.01.
maxPS	A number above which estimated propensity scores should be trimmed; scalar

ipcw.method The censoring model. Allowed values are: 'breslow' (Cox regression with Breslow estimator of the baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)'. Default is 'breslow'.

initial.predictor.method
A character vector for the method used to get initial outcome predictions conditional on the covariates in cate.model in score.method = 'twoReg' and 'contrastReg'. Allowed values include one of 'randomForest', 'boosting' and 'logistic' (fastest). Default is 'randomForest'.

tree.depth A positive integer specifying the depth of individual trees in boosting (usually 2-3). Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is 3.

n.trees.rf A positive integer specifying the maximum number of trees in random forest. Used if score.method = 'ranfomForest' or if initial.predictor.method = 'randomForest' with score.method = 'twoReg' or 'contrastReg'. Default is 1000.

n.trees.boosting
A positive integer specifying the maximum number of trees in boosting (usually 100-1000). Used if score.method = 'boosting' or if initial.predictor.method = 'boosting' with score.method = 'twoReg' or 'contrastReg'. Default is 150.

B A positive integer specifying the number of time cross-fitting is repeated in score.method = 'twoReg' and 'contrastReg'. Default is 3.

Kfold A positive integer specifying the number of folds (parts) used in cross-fitting to partition the data in score.method = 'twoReg' and 'contrastReg'. Default is 5.

plot.gbmperf A logical value indicating whether to plot the performance measures in boosting. Used only if score.method = 'boosting' or if score.method = 'twoReg' or 'contrastReg' and initial.predictor.method = 'boosting'. Default is TRUE.

error.maxNR A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.

max.iterNR A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 100.

tune A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

... Additional arguments for gbm()

Value

Depending on what score.method is, the outputs is a combination of the following: result.randomForest: Results of random forest fit, for trt = 0 and trt = 1 separately result.boosting: Results of boosting fit,

for trt = 0 and trt = 1 separately result.poisson: Naive Poisson estimator (beta1 - beta0); vector of length p.cate + 1 result.twoReg: Two regression estimator (beta1 - beta0); vector of length p.cate + 1 result.contrastReg: A list of the contrast regression results with 2 elements: $delta.contrastReg: Contrast regression DR estimator; vector of length p.cate + 1 $converge.contrastReg: Indicator that the Newton Raphson algorithm converged for delta_0; boolean

---

`ipcw.surv` *Probability of being censored*

Description

Probability of being censored which is used to correct the effect of right censoring.

Usage

```
ipcw.surv(
  y,
  d,
  x.ipcw,
  yf = NULL,
  ipcw.method = "breslow",
  tau0,
  surv.min = 0.025
)
```

Arguments

y	Observed survival or censoring time; vector of size n.
d	The event indicator, normally 1 = event, 0 = censored; vector of size n.
x.ipcw	Matrix of p.ipw baseline covariate specified in inverse probability of censoring weighting; dimension n by p.ipw.
yf	Follow-up time, interpreted as the potential censoring time; vector of size n if the potential censoring time is known. If unknown, set yf == NULL and yf will be taken as y in the function.
ipcw.method	The censoring model. Allowed values are: 'breslow' (Cox regression with Breslow estimator of the baseline survivor function), 'aft (exponential)', 'aft (weibull)', 'aft (lognormal)' or 'aft (loglogistic)'. Default is 'breslow'.
tau0	The truncation time for defining restricted mean time lost.
surv.min	Lower truncation limit for probability of being censored (positive and very close to 0).

Value

A vector of size n with the estimated probabilities Pr(C > min(y, tau0) | x.ipcw)

meanCatch *Catch errors and warnings when estimating the ATEs in the nested subgroup for continuous data*

Description

Storing the errors and warnings that occurred when estimating the ATEs in the nested subgroups. If there are no errors and no warnings, the estimated mean difference is provided. If there are warnings but no errors, the estimated mean difference is provided with a warning attribute set. If there are errors, the NA values are returned for mean difference. A error attribute set is also provided.

Usage

```
meanCatch(fun)
```

Arguments

fun	The drsurv function...

Value

A list containing

meanExample *Simulated data with a continuous outcome*

Description

A dataset containing a continuous outcome and 6 baseline covariates

Usage

```
data(meanExample)
```

Format

A dataframe with 4000 rows (patients) and 9 variables:

age age at baseline, centered to 48 years old, in years

female sex, 0 for male, 1 for female

previous_treatment previous treatment, "drugA", "drugB", or "drugC"

previous_cost previous medical cost, in US dollars

previous_number_symptoms previous number of symptoms, "0", "1", or ">=2"

previous_number_relapses previous number of relapses

trt current treatment, "drug0" or "drug1"

y count outcome, current number of relapses#'

Examples

```
data(meanExample)
str(meanExample)
```

`onearmglmcount.dr` *Doubly robust estimators of the coefficients in the two regression*

Description

Doubly robust estimators of the coefficients in the two regression

Usage

```
onearmglmcount.dr(y, x.cate, time, trt, ps, f.predictor)
```

Arguments

`y`	Observed outcome; vector of size `n`
`x.cate`	Matrix of `p` baseline covariates; dimension `n` by `p`
`time`	Log-transformed person-years of follow-up; vector of size `n`
`trt`	Treatment received; vector of size `n` units with treatment coded as 0/1
`ps`	Estimated propensity scores for all observations; vector of size `n`
`f.predictor`	Initial prediction of the outcome (expected number of relapses for one unit of exposure time) conditioned on the covariates `x` for one treatment group `r`; `mu_r(x)`, step 1 in the two regression; vector of size `n`

Value

Doubly robust estimators of the regression coefficients `beta_r` in the doubly robust estimating equation where `r = 0, 1` is treatment received; vector of size `p + 1` (intercept included)

`onearmglmmean.dr` *Doubly robust estimators of the coefficients in the two regression*

Description

Doubly robust estimators of the coefficients in the two regression

Usage

```
onearmglmmean.dr(y, x.cate, trt, ps, f.predictor)
```

Arguments

y	Observed outcome; vector of size n
x.cate	Matrix of p baseline covariates; dimension n by p
trt	Treatment received; vector of size n units with treatment coded as 0/1
ps	Estimated propensity scores for all observations; vector of size n
f.predictor	Initial prediction of the outcome (expected number of relapses for one unit of exposure time) conditioned on the covariates x for one treatment group r; mu_r(x), step 1 in the two regression; vector of size n

Value

Doubly robust estimators of the regression coefficients beta_r in the doubly robust estimating equation where r = 0, 1 is treatment received; vector of size p + 1 (intercept included)

---

onearmsurv.dr *Doubly robust estimators of the coefficients in the two regression*

Description

Doubly robust estimators of the coefficients in the two regression

Usage

```
onearmsurv.dr(ynew, dnew, trt, x.cate, tau0, weightsurv, ps, f.predictor)
```

Arguments

ynew	Truncated survival or censoring time; vector of size n.
dnew	The event indicator after truncation, 1 = event or censored after truncation, 0 = censored before truncation; vector of size n.
trt	Treatment received; vector of size n with treatment coded as 0/1.
x.cate	Matrix of p.cate baseline covariates specified in the outcome model; dimension n by p.cate.
tau0	The truncation time for defining restricted mean time lost.
weightsurv	Estimated inverse probability of censoring weights with truncation for all observations; vector of size n.
ps	Estimated propensity scores for all observations; vector of size n
f.predictor	Initial prediction of the outcome (restricted mean time loss) conditioned on the covariates x.cate for one treatment group r; mu_r(x.cate), step 1 in the two regression; vector of size n

Value

Doubly robust estimators of the two regression coefficients beta_r where r = 0, 1 is treatment received; vector of size p.cate + 1 (intercept included)

plot.atefit *Histogram of bootstrap estimates*

Description

Histogram of bootstrap estimates

Usage

```
## S3 method for class 'atefit'
plot(x, bins, alpha = 0.7, title = waiver(), theme = theme_classic(), ...)
```

Arguments

x	An object of class "atefit".
bins	Number of bins
alpha	Opacity
title	The text for the title
theme	Defaults to theme_classic(). Other options include theme_grey(), theme_bw(), theme_light(), theme_dark(), and theme_void()
...	Other parameters

Details

Create a histogram displaying the distribution of the bootstrap estimates. The red vertical reference line represents the final estimate.

Value

A plot of the class ggplot, displaying the estimated ATE across the bootstrap samples

Author(s)

Thomas Debray

plot.precmed *Two side-by-side line plots of validation curves from the* "precmed" *object*

Description

Provides validation curves in two side-by-side plots, visualizing the estimated ATEs in a series of nested subgroups in the training set and validation set separately, where each line represents one scoring method specified in catecv() or catecvmean(). This should be run only after results of catecv() or catecvmean() have been obtained.

Usage

```
## S3 method for class 'precmed'
plot(
  x,
  cv.i = NULL,
  combine = "mean",
  show.abc = TRUE,
  valid.only = FALSE,
  plot.hr = FALSE,
  ylab = NULL,
  legend.position = "bottom",
  xlim = NULL,
  title = waiver(),
  theme = theme_classic(),
  ...
)
```

Arguments

x	An object of class "precmed".
cv.i	A positive integer indicating the index of the CV iteration results to be plotted. Allowed values are: a positive integer $<=$ cv.n in catecv() or NULL. If cv.i = NULL, the results across all CV iterations are combined according to combine and then plotted. Default is NULL.
combine	A character value indicating how to combine the estimated ATEs across all CV iterations into a validation curve for each nested subgroup, separately for the training and validation results. Allowed values are: 'mean' or 'median'. Used only if cv.i = NULL. Default is 'mean'.
show.abc	A logical value indicating whether to show the ABC statistics in the validation set. Used only if x$abc = TRUE and xlim is not limited to a smaller range (i.e., xlim = NULL or equal to the entire x$prop.onlyhigh range). If cv.i is NULL, ABC statistics will be based on the combined CV iterations. If cv.i is an integer, ABC statistics will be based solely on that CV iteration. Default is TRUE.

valid.only	A logical value indicating whether only the validation curves in the validation set should be plotted (TRUE). Otherwise, the validation curves in both the training and validation sets are plotted side-by-side (FALSE). Default is FALSE.
plot.hr	A logical value indicating whether the hazard ratios should be plotted in the validation curves (TRUE). Otherwise, the restricted mean time lost is plotted (FALSE). This argument is only applicable to survival outcomes. Default is FALSE.
ylab	A character value for the y-axis label to describe what the ATE is. Default is NULL, which creates a default y-axis label based on available data.
legend.position	A character value for the legend position argument to be passed to ggplot object. Default is 'bottom'.
xlim	A numeric value for the range of the x-axis. Default is NULL, which means there is no range specified.
title	The text for the title
theme	Defaults to theme_classic(). Other options include theme_grey(), theme_bw(), theme_light(), theme_dark(), and theme_void()
...	Other parameters

Details

plot() takes in outputs from catecv() and generates two plots of validation curves side-by-side, one for the training set and one for validation set. Separate validation curves are produced for each scoring method specified via score.method in catecv() or catecvmean().

The validation curves (and ABC statistics, if applicable) can help compare the performance of different scoring methods in terms of discerning potential treatment heterogeneity in subgroups with internal validation. Steeper validation curves in the validation set suggest presence of treatment effect heterogeneity (and the ability of the scoring methods to capture it) while flat validation curves indicate absence of treatment effect heterogeneity (or inability of the scoring method to capture it).

Value

Returns two side-by-side line plots, one of which shows the validation curves of the training sets and the other the validation curves in the validation sets. A gray horizontal dashed line of overall ATE is included as a reference. ABC statistics will be added to the legend if show.abc = TRUE.

References

Yadlowsky, S., Pellegrini, F., Lionetto, F., Braune, S., & Tian, L. (2020). *Estimation and validation of ratio-based conditional average treatment effects using observational data. Journal of the American Statistical Association, 1-18.* DOI: 10.1080/01621459.2020.1772080.

See Also

abc() and boxplot() for "precmed" objects.

Examples

```
# Count outcome
eval_1 <- catecv(response = "count",
                 data = countExample,
                 score.method = "poisson",
                 cate.model = y ~ age + female + previous_treatment +
                           previous_cost + previous_number_relapses + offset(log(years)),
                 ps.model = trt ~ age + previous_treatment,
                 higher.y = FALSE,
                 cv.n = 5)

# default setting
plot(eval_1)

# turn off ABC annotation
plot(eval_1, show.abc = FALSE)

# use a different theme
plot(eval_1, theme = ggplot2::theme_bw())

# plot the validation curves from the 2nd CV iteration instead of the mean
# of all validation curves
plot(eval_1, cv.i = 2)

# median of the validation curves
plot(eval_1, combine = "median")

# plot validation curves in validation set only
plot(eval_1, valid.only = TRUE)

# Survival outcome
library(survival)
tau0 <- with(survivalExample,
             min(quantile(y[trt == "drug1"], 0.95), quantile(y[trt == "drug0"], 0.95)))
eval_2 <- catecv(response = "survival",
                 data = survivalExample,
                 score.method = c("poisson", "randomForest"),
                 cate.model = Surv(y, d) ~ age + female + previous_cost +
                                           previous_number_relapses,
                 ps.model = trt ~ age + previous_treatment,
                 initial.predictor.method = "randomForest",
                 ipcw.model = ~ age + previous_cost + previous_treatment,
                 tau0 = tau0,
                 cv.n = 5,
                 seed = 999)


# default setting, plot RMTL ratios in both training and validation sets
plot(eval_2)

# plot hazard ratio
plot(eval_2, plot.hr = TRUE)
```

print.atefit *Print function for atefit*

Description

Print function for atefit

Usage

```
## S3 method for class 'atefit'
print(x, ...)
```

Arguments

x	An object of class "atefit".
...	Other parameters

Details

Display the estimated treatment effects for survival outcomes (log restricted mean time lost ratio and log hazard ratio) and count outcomes (the log rate ratio).

Value

No return value

Author(s)

Thomas Debray

print.catefit *Print function for atefit*

Description

Print function for atefit

Usage

```
## S3 method for class 'catefit'
print(x, ...)
```

Arguments

x	An object of class "catefit".
...	Other parameters

Details

Display the estimated treatment effects for survival outcomes (log restricted mean time lost ratio and log hazard ratio) and count outcomes (the log rate ratio).

Value

No return value

Author(s)

Thomas Debray

---

scorecount *Calculate the log CATE score given the baseline covariates and follow-up time for specified scoring method methods*

Description

Based on intxcount results of the CATE coefficients estimated with boosting, naive Poisson, two regression, contrast regression, negative binomial

Usage

```
scorecount(
  fit,
  x.cate,
  time,
  score.method = c("boosting", "poisson", "twoReg", "contrastReg", "negBin")
)
```

Arguments

fit	List of objects generated from intxcount: outputs of boosting, naive Poisson, two regression, contrast regression, negative binomial
x.cate	Matrix of p.cate baseline covariates; dimension n (observations) by p.cate (covariates in the outcome model)
time	Log-transformed person-years of follow-up; vector of size n
score.method	A vector of one or multiple methods to estimate the CATE score. Allowed values are: 'boosting', 'poisson', 'twoReg', 'contrastReg', 'negBin'. Default specifies all 5 methods.

Value

score.boosting: Estimated log CATE score for all n observations with the boosting method; vector of size n score.poisson: Estimated log CATE score for all n observations with the naive Poisson method; vector of size n score.twoReg: Estimated log CATE score for all n observations with the two regression method; vector of size n score.contrastReg: Estimated log CATE score for all n observations with the contrast regression method; vector of size n score.negBin: Estimated log CATE score for all n observations with the naive Poisson method; vector of size n score = NA if the corresponding method is not called

---

`scoremean` *Calculate the CATE score given the baseline covariates for specified scoring method methods*

Description

Based on intxmean results of the CATE coefficients estimated with boosting, linear regression, two regression, contrast regression, random forest, generalized additive model

Usage

```
scoremean(
  fit,
  x.cate,
 score.method = c("boosting", "gaussian", "twoReg", "contrastReg", "randomForest",
    "gam")
)
```

Arguments

`fit`	List of objects generated from intxmean: outputs of boosting, linear regression, two regression, contrast regression, random forest, generalized additive model
`x.cate`	Matrix of `p.cate` baseline covariates; dimension `n` (observations) by `p.cate` (covariates in the outcome model)
`score.method`	A vector of one or multiple methods to estimate the CATE score. Allowed values are: `'boosting'`, `'gaussian'`, `'twoReg'`, `'contrastReg'`, `'randomForest'`, `'gam'`. Default specifies all 6 methods.

Value

score.boosting: Estimated CATE score for all n observations with the boosting method; vector of size n score.gaussian: Estimated CATE score for all n observations with the linear regression method; vector of size n score.twoReg: Estimated CATE score for all n observations with the two regression method; vector of size n score.contrastReg: Estimated CATE score for all n observations with the contrast regression method; vector of size n score.randomForest: Estimated CATE score for all n observations with the random forest method; vector of size n score.gam: Estimated CATE score for all n observations with the generalized additive model; vector of size n score = NA if the corresponding method is not called

`scoresurv` *Calculate the log CATE score given the baseline covariates and follow-up time for specified scoring method methods for survival outcomes*

Description

Based on intxsurv results of the CATE coefficients estimated with random forest, boosting, naive Poisson, two regression, contrast regression

Usage

```
scoresurv(
  fit,
  x.cate,
  tau0,
 score.method = c("randomForest", "boosting", "poisson", "twoReg", "contrastReg")
)
```

Arguments

`fit`	List of objects generated from intxsurv: outputs of random forest, boosting, naive Poisson, two regression, contrast regression
`x.cate`	Matrix of `p.cate` baseline covariates specified in the outcome model; dimension `n` by `p.cate`.
`tau0`	The truncation time for defining restricted mean time lost.
`score.method`	A vector of one or multiple methods to estimate the CATE score. Allowed values are: `'randomForest'`, `'boosting'`, `'poisson'`, `'twoReg'`, `'contrastReg'`. Default specifies all 5 methods.

Value

score.randomForest: Estimated log CATE score for all `n` observations with the random forest method; vector of size `n` score.boosting: Estimated log CATE score for all `n` observations with the boosting method; vector of size `n` score.poisson: Estimated log CATE score for all `n` observations with the naive Poisson method; vector of size `n` score.twoReg: Estimated log CATE score for all `n` observations with the two regression method; vector of size `n` score.contrastReg: Estimated log CATE score for all `n` observations with the contrast regression method; vector of size `n` score = NA if the corresponding method is not called

survCatch — *Catch errors and warnings when estimating the ATEs in the nested subgroup*

Description

Storing the errors and warnings that occurred when estimating the ATEs in the nested subgroups. If there are no errors and no warnings, the estimated log.rmtl.ratio and log.hazard.ratio are provided. If there are warnings but no errors, the estimated log.rmtl.ratio and log.hazard.ratio are provided with a warning attribute set. If there are errors, the NA values are returned for log.rmtl.ratio and log.hazard.ratio. A error attribute set is also provided.

Usage

```
survCatch(fun)
```

Arguments

fun	The drsurv function...

Value

A list containing

survivalExample — *Simulated data with survival outcome*

Description

A dataset containing a time-to-event outcome, an event indicator, treatment group, and 6 baseline covariates

Usage

```
data(survivalExample)
```

Format

A dataframe with 4000 rows (patients) and 9 variables:

age age at baseline, centered to 48 years old, in years

female sex, 0 for male, 1 for female

previous_treatment previous treatment, "drugA", "drugB", or "drugC"

previous_cost previous medical cost, in US dollars

previous_number_symptoms previous number of symptoms, "0", "1", or ">=2"

previous_number_relapses previous number of relapses

trt current treatment, "drug0" or "drug1"

y time to first relapse or censoring

d event indicator, 1: relapse, 0: censored

Examples

```
data(survivalExample)
str(survivalExample)
```

`twoarmglmcount.dr` *Doubly robust estimators of the coefficients in the contrast regression as well as their covariance matrix and convergence information*

Description

Newton-Raphson algorithm is used to solve the estimating equation `bar S_n (delta) = 0`

Usage

```
twoarmglmcount.dr(
  y,
  x.cate,
  time,
  trt,
  ps,
  f1.predictor,
  f0.predictor,
  error.maxNR = 0.001,
  max.iterNR = 150,
  tune = c(0.5, 2)
)
```

Arguments

`y`	Observed outcome; vector of size `n`
`x.cate`	Matrix of `p.cate` baseline covariates; dimension `n` by `p.cate`
`time`	Log-transformed person-years of follow-up; vector of size `n`
`trt`	Treatment received; vector of size `n` units with treatment coded as 0/1
`ps`	Estimated propensity scores for all observations; vector of size `n`
`f1.predictor`	Initial predictions of the outcome (expected number of relapses for one unit of exposure time) conditioned on the covariates `x` for treatment group trt = 1; `mu_1(x)`, step 1 in the two regression; vector of size `n`

f0.predictor	Initial predictions of the outcome (expected number of relapses for one unit of exposure time) conditioned on the covariates x for treatment group trt = 0; mu_0(x), step 1 in the two regression; vector of size n
error.maxNR	A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.
max.iterNR	A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 150.
tune	A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

Value

coef: Doubly robust estimators of the regression coefficients delta_0; vector of size p + 1 (intercept included) vcov: Variance-covariance matrix of the estimated coefficient delta_0; matrix of size p + 1 by p + 1 converge: Indicator that the Newton Raphson algorithm converged for delta_0; boolean

---

twoarmglmmean.dr *Doubly robust estimators of the coefficients in the contrast regression as well as their covariance matrix*

Description

Solving the estimating equation bar S_n (delta) = 0

Usage

```
twoarmglmmean.dr(y, x.cate, trt, ps, f1.predictor, f0.predictor)
```

Arguments

y	Observed outcome; vector of size n
x.cate	Matrix of p.cate baseline covariates; dimension n by p.cate
trt	Treatment received; vector of size n units with treatment coded as 0/1
ps	Estimated propensity scores for all observations; vector of size n
f1.predictor	Initial predictions of the outcome (expected number of relapses for one unit of exposure time) conditioned on the covariates x for treatment group trt = 1; mu_1(x), step 1 in the two regression; vector of size n
f0.predictor	Initial predictions of the outcome (expected number of relapses for one unit of exposure time) conditioned on the covariates x for treatment group trt = 0; mu_0(x), step 1 in the two regression; vector of size n

Value

coef: Doubly robust estimators of the regression coefficients `delta_0`; vector of size `p + 1` (intercept included) vcov: Variance-covariance matrix of the estimated coefficient `delta_0`; matrix of size `p + 1` by `p + 1`

`twoarmsurv.dr` *Doubly robust estimators of the coefficients in the contrast regression as well as their covariance matrix and convergence information*

Description

Newton-Raphson algorithm is used to solve the estimating equation `bar S_n (delta) = 0`

Usage

```
twoarmsurv.dr(
  ynew,
  dnew,
  trt,
  x.cate,
  tau0,
  weightsurv,
  ps,
  f1.predictor,
  f0.predictor,
  error.maxNR = 0.001,
  max.iterNR = 100,
  tune = c(0.5, 2)
)
```

Arguments

`ynew`	Truncated survival time; vector of size `n`
`dnew`	Event indicator after truncation; vector of size `n`
`trt`	Treatment received; vector of size `n` with treatment coded as 0/1.
`x.cate`	Matrix of `p.cate` baseline covariates specified in the outcome model; dimension `n` by `p.cate`.
`tau0`	The truncation time for defining restricted mean time lost.
`weightsurv`	Estimated inverse probability of censoring weights with truncation for all observations; vector of size `n`.
`ps`	Estimated propensity scores for all observations; vector of size `n`
`f1.predictor`	Initial predictions of the outcome (restricted mean time loss) conditioned on the covariates `x.cate` for treatment group trt = 1; `mu_1(x.cate)`, step 1 in the two regression; vector of size `n`

f0.predictor	Initial predictions of the outcome (restricted mean time loss) conditioned on the covariates x.cate for treatment group trt = 0; mu_0(x.cate), step 1 in the two regression; vector of size n
error.maxNR	A numerical value > 0 indicating the minimum value of the mean absolute error in Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 0.001.
max.iterNR	A positive integer indicating the maximum number of iterations in the Newton Raphson algorithm. Used only if score.method = 'contrastReg'. Default is 100.
tune	A vector of 2 numerical values > 0 specifying tuning parameters for the Newton Raphson algorithm. tune[1] is the step size, tune[2] specifies a quantity to be added to diagonal of the slope matrix to prevent singularity. Used only if score.method = 'contrastReg'. Default is c(0.5, 2).

Value

coef: Doubly robust estimators of the contrast regression coefficients delta_0; vector of size p.cate + 1 (intercept included) converge: Indicator that the Newton Raphson algorithm converged for delta_0; boolean

Index