Package ‘evalITR’

August 26, 2023

Title Evaluating Individualized Treatment Rules

Version 1.0.0

Date 2023-08-20

Maintainer Michael Lingzhi Li <mili@hbs.edu>


License GPL (>= 2)

URL https://github.com/MichaelLLi/evalITR,
https://michaellli.github.io/evalITR/,
https://jialul.github.io/causal-ml/

BugReports https://github.com/MichaelLLi/evalITR/issues

Depends dplyr (>= 1.0), MASS (>= 7.0), Matrix (>= 1.0), quadprog (>= 1.0), R (>= 3.5.0), stats

Imports caret, cli, e1071, forcats, gbm, ggdist, ggplot2, ggthemes, glamnet, grf, haven, purrr, rlang, rpart, rqPen, scales, utils, bartCause, SuperLearner

Suggests doParallel, furrr, knitr, rmarkdown, testthat, bartMachine, elasticnet, randomForest, spelling

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.2.2

Language en-US

NeedsCompilation no
R topics documented:

AUPEC ...................................................... 3
AUPEcv .................................................. 4
compute_qoi ........................................... 5
calculate_qoi_user ................................. 5
consist.test ................................. 6
consistencycv.test ......................... 7
create_ml_args ............................... 8
create_ml_args_bart .................... 9
create_ml_args_bartc ................. 9
create_ml_args_causalforest ... 9
create_ml_args_lasso .............. 10
create_ml_args_superLearner ... 10
create_ml_args_svm ............... 10
create_ml_args_svm_cls .......... 11
create_ml_arguments ............... 11
estimate_itr ......................... 12
evaluate_itr ..................... 13
fit_itr ........................................... 14
GATE .............................................. 15
GATEcv .......................................... 16
het.test ........................................ 17
hetcv.test ................................... 18
PAPD ............................................. 19
PAPDcv ......................................... 20
PAPE ............................................. 22
PAPEcv ......................................... 23
PAV ............................................... 24
PAVcv ........................................... 25
plot_itr ......................................... 26
plot_estimate ......................... 27
print.summary_itr .................. 27
print.summary.test_itr ....... 28
star ............................................. 28
summary_itr .......................... 29
summary.test_itr ............. 29
test_itr ...................................... 30

Index 31
Description

This function estimates AUPEC. The details of the methods for this design are given in Imai and Li (2019).

Usage

AUPEC(T, tau, Y, centered = TRUE)

Arguments

T A vector of the unit-level binary treatment receipt variable for each sample.

tau A vector of the unit-level continuous score for treatment assignment. We assume those that have tau<0 should not have treatment. Conditional Average Treatment Effect is one possible measure.

Y A vector of the outcome variable of interest for each sample.

centered If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.

Value

A list that contains the following items:

aupec The estimated Area Under Prescription Evaluation Curve

sd The estimated standard deviation of AUPEC.

vec A vector of points outlining the AUPEC curve across each possible budget point for the dataset. Each step increases the budget by 1/n where n is the number of data points.

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/

References

Imai and Li (2019). “Experimental Evaluation of Individualized Treatment Rules”,

Imai and Li (2019).
Examples

```r
T = c(1,0,1,0,1,0,1,0)
tau = c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7)
Y = c(4,5,0,2,4,1,-4,3)
aupeclist <- AUPEC(T,tau,Y)
aupeclist$aupec
aupeclist$sd
aupeclist$vec
```

AUPECcv

*Estimation of the Area Under Prescription Evaluation Curve (AUPEC) in Randomized Experiments Under Cross Validation*

Description

This function estimates AUPEC. The details of the methods for this design are given in Imai and Li (2019).

Usage

```r
AUPECcv(T, tau, Y, ind, centered = TRUE)
```

Arguments

- **T**: A vector of the unit-level binary treatment receipt variable for each sample.
- **tau**: A matrix where the $i$th column is the unit-level continuous score for treatment assignment generated in the $i$th fold.
- **Y**: The outcome variable of interest.
- **ind**: A vector of integers (between 1 and number of folds inclusive) indicating which testing set does each sample belong to.
- **centered**: If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.

Value

A list that contains the following items:

- **aupec**: The estimated AUPEC.
- **sd**: The estimated standard deviation of AUPEC.

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;
References


Examples

\[ T = c(1,0,1,0,1,0,1,0) \]
\[ \tau = \text{matrix}(c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,-0.5,-0.3,-0.1,0.1,0.3,0.5,0.7,0.9), nrow = 8, ncol = 2) \]
\[ Y = c(4,5,0,2,4,1,-4,3) \]
\[ \text{ind} = \text{c(rep}(1,4),\text{rep}(2,4)) \]
\[ \text{aupeclist} \leftarrow \text{AUPECcv}(T, \tau, Y, \text{ind}) \]
\[ \text{aupeclist}$$aupec \]
\[ \text{aupeclist}$sd \]

---

**compute_qoi**

*Compute Quantities of Interest (PAPE, PAPEp, PAPDp, AUPEC, GATE, GATEcv)*

---

**Description**

Compute Quantities of Interest (PAPE, PAPEp, PAPDp, AUPEC, GATE, GATEcv)

**Usage**

```r
compute_qoi(fit_obj, algorithms)
```

**Arguments**

- `fit_obj`: An output object from `fit_itr` function.
- `algorithms`: Machine learning algorithms

---

**compute_qoi_user**

*Compute Quantities of Interest (PAPE, PAPEp, PAPDp, AUPEC, GATE, GATEcv) with user defined functions*

---

**Description**

Compute Quantities of Interest (PAPE, PAPEp, PAPDp, AUPEC, GATE, GATEcv) with user defined functions

**Usage**

```r
compute_qoi_user(user_itr, Tcv, Ycv, data, ngates, budget, ...)
```
**Arguments**

- **user_itr**: A user-defined function to create an ITR. The function should take the data as input and return an unit-level continuous score for treatment assignment. We assume those that have score less than 0 should not have treatment. The default is `NULL`, which means the ITR will be estimated from the `estimate_itr`.

- **Tcv**: A vector of the unit-level binary treatment.

- **Ycv**: A vector of the unit-level continuous outcome.

- **data**: A data frame containing the variables of interest.

- **ngates**: The number of gates to be used in the GATE function.

- **budget**: The maximum percentage of population that can be treated under the budget constraint.

- **...**: Additional arguments to be passed to the user-defined function.

---

**consist.test**

*The Consistency Test for Grouped Average Treatment Effects (GATEs) in Randomized Experiments*

**Description**

This function calculates statistics related to the test of treatment effect consistency across groups.

**Usage**

```
consist.test(T, tau, Y, ngates = 5, nsim = 10000)
```

**Arguments**

- **T**: A vector of the unit-level binary treatment receipt variable for each sample.

- **tau**: A vector of the unit-level continuous score. Conditional Average Treatment Effect is one possible measure.

- **Y**: A vector of the outcome variable of interest for each sample.

- **ngates**: The number of groups to separate the data into. The groups are determined by `tau`. Default is 5.

- **nsim**: Number of Monte Carlo simulations used to simulate the null distributions. Default is 10000.

**Details**

The details of the methods for this design are given in Imai and Li (2022).

**Value**

A list that contains the following items:

- **stat**: The estimated statistic for the test of consistency

- **pval**: The p-value of the null hypothesis (that the treatment effects are consistent)
Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;

References

Imai and Li (2022). “Statistical Inference for Heterogeneous Treatment Effects Discovered by Generic Machine Learning in Randomized Experiments”.

Examples

```r
T = c(1,0,1,0,1,0,1,0)
tau = c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7)
Y = c(4,5,0,2,4,1,-4,3)
consisttestlist <- consist.test(T,tau,Y,ngates=5)
consisttestlist$stat
consisttestlist$pval
```

Description

This function calculates statistics related to the test of treatment effect consistency across groups under cross-validation.

Usage

```r
consistcv.test(T, tau, Y, ind, ngates = 5, nsim = 10000)
```

Arguments

- `T`: A vector of the unit-level binary treatment receipt variable for each sample.
- `tau`: A vector of the unit-level continuous score. Conditional Average Treatment Effect is one possible measure.
- `Y`: A vector of the outcome variable of interest for each sample.
- `ind`: A vector of integers (between 1 and number of folds inclusive) indicating which testing set does each sample belong to.
- `ngates`: The number of groups to separate the data into. The groups are determined by `tau`. Default is 5.
- `nsim`: Number of Monte Carlo simulations used to simulate the null distributions. Default is 10000.

Details

The details of the methods for this design are given in Imai and Li (2022).
create_ml_args

Value

A list that contains the following items:

stat
The estimated statistic for the test of consistency under cross-validation.
pval
The p-value of the null hypothesis (that the treatment effects are consistent)

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;

References

Imai and Li (2022). “Statistical Inference for Heterogeneous Treatment Effects Discovered by Generic Machine Learning in Randomized Experiments”.

Examples

T = c(1,0,1,0,1,0,1,0)
tau = matrix(c(0,0,1,0,2,0,3,0,4,0,5,0,6,0,7,-0.5,-0.3,-0.1,0,1,0.3,0.5,0.7,0.9),nrow = 8, ncol = 2)
Y = c(4,5,0,2,4,1,-4,3)
ind = c(rep(1,4),rep(2,4))
consisttestlist <- consistcv.test(T,tau,Y,ind,ngates=2)
consisttestlist$stat
consisttestlist$pval

create_ml_args

Create general arguments

Description

Create general arguments

Usage

create_ml_args(data)

Arguments

data A dataset
create_ml_args_bart  
Create arguments for bartMachine

**Description**
Create arguments for bartMachine

**Usage**
create_ml_args_bart(data)

**Arguments**
- **data**  
  A dataset

create_ml_args_bartc  
Create arguments for bartCause

**Description**
Create arguments for bartCause

**Usage**
create_ml_args_bartc(data)

**Arguments**
- **data**  
  A dataset

create_ml_args_causalforest  
Create arguments for causal forest

**Description**
Create arguments for causal forest

**Usage**
create_ml_args_causalforest(data)

**Arguments**
- **data**  
  A dataset
create_ml_args_lasso  
*Create arguments for LASSO*

**Description**
Create arguments for LASSO

**Usage**
create_ml_args_lasso(data)

**Arguments**
- data: A dataset

create_ml_args_superLearner  
*Create arguments for super learner*

**Description**
Create arguments for super learner

**Usage**
create_ml_args_superLearner(data)

**Arguments**
- data: A dataset

create_ml_args_svm  
*Create arguments for SVM*

**Description**
Create arguments for SVM

**Usage**
create_ml_args_svm(data)

**Arguments**
- data: A dataset
create_ml_args_svm_cls

Create arguments for SVM classification

Description
Create arguments for SVM classification

Usage
create_ml_args_svm_cls(data)

Arguments
- data: A dataset

create_ml_arguments
Create arguments for ML algorithms

Description
Create arguments for ML algorithms

Usage
create_ml_arguments(outcome, treatment, data)

Arguments
- outcome: Outcome of interests
- treatment: Treatment variable
- data: A dataset
estimate_itr  

Estimate individual treatment rules (ITR)

Description

Estimate individual treatment rules (ITR)

Usage

```
estimate_itr(
  treatment,
  form,
  data,
  algorithms,
  budget,
  n_folds = 5,
  split_ratio = 0,
  ngates = 5,
  preProcess = NULL,
  weights = NULL,
  trControl = caret::trainControl(method = "none"),
  tuneGrid = NULL,
  tuneLength = ifelse(trControl$method == "none", 1, 3),
  user_model = NULL,
  SL_library = NULL,
  ...
)
```

Arguments

- **treatment**: Treatment variable
- **form**: a formula object that takes the form `y ~ T + x1 + x2 + ...`
- **data**: A data frame that contains the outcome `y` and the treatment `T`.
- **algorithms**: List of machine learning algorithms to be used.
- **budget**: The maximum percentage of population that can be treated under the budget constraint.
- **n_folds**: Number of cross-validation folds. Default is 5.
- **split_ratio**: Split ratio between train and test set under sample splitting. Default is 0.
- **ngates**: The number of groups to separate the data into. The groups are determined by tau. Default is 5.
- **preProcess**: caret parameter
- **weights**: caret parameter
- **trControl**: caret parameter
- **tuneGrid**: caret parameter
- **tuneLength**: caret parameter
- **user_model**: caret parameter
- **SL_library**: caret parameter
evaluate_itr

<table>
<thead>
<tr>
<th>tunelength</th>
<th>caret parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_model</td>
<td>A user-defined function to create an ITR. The function should take the data as input and return a model to estimate the ITR.</td>
</tr>
<tr>
<td>SL_library</td>
<td>A list of machine learning algorithms to be used in the super learner.</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments passed to caret::train</td>
</tr>
</tbody>
</table>

Value

An object of itr class

evaluate_itr | Evaluate ITR

Description

Evaluate ITR

Usage

```
evaluate_itr(
  fit = NULL,
  user_itr = NULL,
  outcome = c(),
  treatment = c(),
  data = list(),
  budget = 1,
  ngates = 5,
  ...
)
```

Arguments

- **fit** Fitted model. Usually an output from estimate_itr
- **user_itr** A user-defined function to create an ITR. The function should take the data as input and return an unit-level continuous score for treatment assignment. We assume those that have score less than 0 should not have treatment. The default is NULL, which means the ITR will be estimated from the estimate_itr.
- **outcome** A character string of the outcome variable name.
- **treatment** A character string of the treatment variable name.
- **data** A data frame containing the variables specified in outcome, treatment, and tau.
- **budget** The maximum percentage of population that can be treated under the budget constraint.
ngates The number of gates to use for the ITR. The default is 5. A user-defined function to create an ITR. The function should take the data as input and return an ITR. The output is a vector of the unit-level binary treatment that would have been assigned by the individualized treatment rule. The default is NULL, which means the ITR will be estimated from the estimate_itr. See ?evaluate_itr for an example.

... Further arguments passed to the function.

Value

An object of itr class

fit_itr Estimate ITR for Single Outcome

Description

Estimate ITR for Single Outcome

Usage

fit_itr(data, algorithms, params, folds, budget, user_model, ...)

Arguments

data A dataset.
algorithms Machine learning algorithms.
params A list of parameters.
folds Number of folds.
budget The maximum percentage of population that can be treated under the budget constraint.
user_model User's own function to estimated the ITR.
... Additional arguments passed to caret::train

Value

A list of estimates.
Estimation of the Grouped Average Treatment Effects (GATEs) in Randomized Experiments

Description

This function estimates the Grouped Average Treatment Effects (GATEs) where the groups are determined by a continuous score. The details of the methods for this design are given in Imai and Li (2022).

Usage

GATE(T, tau, Y, ngates = 5)

Arguments

T  A vector of the unit-level binary treatment receipt variable for each sample.

tau  A vector of the unit-level continuous score. Conditional Average Treatment Effect is one possible measure.

Y  A vector of the outcome variable of interest for each sample.

ngates  The number of groups to separate the data into. The groups are determined by tau. Default is 5.

Value

A list that contains the following items:

* gate  The estimated vector of GATEs of length ngates arranged in order of increasing tau.

* sd  The estimated vector of standard deviation of GATEs.

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;

References

Imai and Li (2022). “Statistical Inference for Heterogeneous Treatment Effects Discovered by Generic Machine Learning in Randomized Experiments”,
Examples

```r
t = c(1,0,1,0,1,0,1,0)
tau = c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7)
Y = c(4,5,0,2,4,1,-4,3)
gatelist <- GATE(T,tau,Y,ngates=5)
gatelist$gate
gatelist$sd
```

Description

This function estimates the Grouped Average Treatment Effects (GATEs) under cross-validation where the groups are determined by a continuous score. The details of the methods for this design are given in Imai and Li (2022).

Usage

```r
GATEcv(T, tau, Y, ind, ngates = 5)
```

Arguments

- **T**: A vector of the unit-level binary treatment receipt variable for each sample.
- **tau**: A matrix where the $i$th column is the unit-level continuous score for treatment assignment generated in the $i$th fold. Conditional Average Treatment Effect is one possible measure.
- **Y**: A vector of the outcome variable of interest for each sample.
- **ind**: A vector of integers (between 1 and number of folds inclusive) indicating which testing set does each sample belong to.
- **ngates**: The number of groups to separate the data into. The groups are determined by tau. Default is 5.

Value

A list that contains the following items:

- **gate**: The estimated vector of GATEs under cross-validation of length ngates arranged in order of increasing tau.
- **sd**: The estimated vector of standard deviation of GATEs under cross-validation.

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;
het.test

References
Imai and Li (2022). “Statistical Inference for Heterogeneous Treatment Effects Discovered by Generic Machine Learning in Randomized Experiments”,

Examples
T = c(1,0,1,0,1,0,1,0)
tau = matrix(c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,-0.5,-0.3,-0.1,0.1,0.3,0.5,0.7,0.9),nrow = 8, ncol = 2)
Y = c(4,5,0,2,4,1,-4,3)
ind = c(rep(1,4),rep(2,4))
gatelist <- GATEcv(T, tau, Y, ind, ngates = 2)
gatelist$gate
gatelist.sd

het.test The Heterogeneity Test for Grouped Average Treatment Effects (GATEs) in Randomized Experiments

Description
This function calculates statistics related to the test of heterogeneous treatment effects across groups.

Usage
het.test(T, tau, Y, ngates = 5)

Arguments
T A vector of the unit-level binary treatment receipt variable for each sample.
tau A vector of the unit-level continuous score. Conditional Average Treatment Effect is one possible measure.
Y A vector of the outcome variable of interest for each sample.
ngates The number of groups to separate the data into. The groups are determined by tau. Default is 5.

Details
The details of the methods for this design are given in Imai and Li (2022).

Value
A list that contains the following items:
stat The estimated statistic for the test of heterogeneity.
pval The p-value of the null hypothesis (that the treatment effects are homogeneous)
Author(s)
Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;

References
Imai and Li (2022). “Statistical Inference for Heterogeneous Treatment Effects Discovered by Generic Machine Learning in Randomized Experiments”.

Examples
```r
T = c(1,0,1,0,1,0,1,0)
tau = c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7)
Y = c(4,5,0,2,4,1,-4,3)
hetestlist <- het.test(T,tau,Y,ngates=5)
hetestlist$stat
hetestlist$pval
```

hetcv.test
The Heterogeneity Test for Grouped Average Treatment Effects (GATEs) under Cross Validation in Randomized Experiments

Description
This function calculates statistics related to the test of heterogeneous treatment effects across groups under cross-validation.

Usage
```r
hetcv.test(T, tau, Y, ind, ngates = 5)
```

Arguments
- `T`: A vector of the unit-level binary treatment receipt variable for each sample.
- `tau`: A vector of the unit-level continuous score. Conditional Average Treatment Effect is one possible measure.
- `Y`: A vector of the outcome variable of interest for each sample.
- `ind`: A vector of integers (between 1 and number of folds inclusive) indicating which testing set does each sample belong to.
- `ngates`: The number of groups to separate the data into. The groups are determined by tau. Default is 5.

Details
The details of the methods for this design are given in Imai and Li (2022).
Value

A list that contains the following items:

stat  The estimated statistic for the test of heterogeneity under cross-validation.
pval  The p-value of the null hypothesis (that the treatment effects are homogeneous)

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>,
https://www.michaellz.com/;

References

Imai and Li (2022). “Statistical Inference for Heterogeneous Treatment Effects Discovered by
Generic Machine Learning in Randomized Experiments”.

Examples

T = c(1,0,1,0,1,0,1,0)
tau = matrix(c(0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,-0.5,-0.3,-0.1,0.1,0.3,0.5,0.7,0.9),nrow = 8, ncol = 2)
Y = c(4,5,0,2,4,1,-4,3)
ind = c(rep(1,4),rep(2,4))
hettestlist <- hetcv.test(T,tau,Y,ind,ngates=2)
hettestlist$stat
hettestlist$pval

Description

This function estimates the Population Average Prescription Difference with a budget constraint. The
details of the methods for this design are given in Imai and Li (2019).

Usage

PAPD(T, Thatfp, Thatgp, Y, budget, centered = TRUE)

Arguments

T  A vector of the unit-level binary treatment receipt variable for each sample.
Thatfp  A vector of the unit-level binary treatment that would have been assigned by the
first individualized treatment rule. Please ensure that the percentage of treatment
units of That is lower than the budget constraint.
Thatgp  A vector of the unit-level binary treatment that would have been assigned by
the second individualized treatment rule. Please ensure that the percentage of
treatment units of That is lower than the budget constraint.
Y: A vector of the outcome variable of interest for each sample.
budget: The maximum percentage of population that can be treated under the budget constraint. Should be a decimal between 0 and 1.
centered: If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.

Value

A list that contains the following items:

papd: The estimated Population Average Prescription Difference
sd: The estimated standard deviation of PAPD.

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;

References

Imai and Li (2019). “Experimental Evaluation of Individualized Treatment Rules”,

Examples

T = c(1,0,1,0,1,0,1,0)
That = c(0,1,1,0,0,1,1,0)
That2 = c(1,0,0,1,1,0,0,1)
Y = c(4,5,0,2,4,1,-4,3)
papdlist <- PAPD(T,That,That2,Y,budget = 0.5)
papdlist$papd
papdlist$sd

PAPDcv: Estimation of the Population Average Prescription Difference in Randomized Experiments Under Cross Validation

Description

This function estimates the Population Average Prescription Difference with a budget constraint under cross validation. The details of the methods for this design are given in Imai and Li (2019).

Usage

PAPDcv(T, Thatfp, Thatgp, Y, ind, budget, centered = TRUE)
Arguments

T  A vector of the unit-level binary treatment receipt variable for each sample.

Thatfp  A matrix where the i-th column is the unit-level binary treatment that would have been assigned by the first individualized treatment rule generated in the i-th fold. Please ensure that the percentage of treatment units of That is lower than the budget constraint.

Thatgp  A matrix where the i-th column is the unit-level binary treatment that would have been assigned by the second individualized treatment rule generated in the i-th fold. Please ensure that the percentage of treatment units of That is lower than the budget constraint.

Y  The outcome variable of interest.

ind  A vector of integers (between 1 and number of folds inclusive) indicating which testing set does each sample belong to.

budget  The maximum percentage of population that can be treated under the budget constraint. Should be a decimal between 0 and 1.

centered  If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.

Value

A list that contains the following items:

papd  The estimated Population Average Prescription Difference.

sd  The estimated standard deviation of PAPD.

Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/

References


Examples

\[
\begin{align*}
T & = c(1,0,1,0,1,0,1,0) \\
That & = \text{matrix}(c(0,1,1,0,0,1,1,0,0,0,1,0,1,0,0,1), \text{nrow} = 8, \text{ncol} = 2) \\
That2 & = \text{matrix}(c(0,0,1,1,0,0,1,1,1,0,0,1,1,0,0,1), \text{nrow} = 8, \text{ncol} = 2) \\
Y & = c(4,5,0,2,4,1,-4,3) \\
ind & = \text{c(rep(1,4),rep(2,4))} \\
\text{papdlist} & \leftarrow \text{PAPDcv}(T, \text{That}, \text{That2}, Y, \text{ind}, \text{budget} = 0.5) \\
\text{papdlist}\text{papd} \\
\text{papdlist}\text{sd}
\end{align*}
\]
### Description

This function estimates the Population Average Prescription Effect with and without a budget constraint. The details of the methods for this design are given in Imai and Li (2019).

### Usage

```r
PAPE(T, That, Y, budget = NA, centered = TRUE)
```

### Arguments

- `T`: A vector of the unit-level binary treatment receipt variable for each sample.
- `That`: A vector of the unit-level binary treatment that would have been assigned by the individualized treatment rule. If `budget` is specified, please ensure that the percentage of treatment units of `That` is lower than the budget constraint.
- `Y`: A vector of the outcome variable of interest for each sample.
- `budget`: The maximum percentage of population that can be treated under the budget constraint. Should be a decimal between 0 and 1. Default is NA which assumes no budget constraint.
- `centered`: If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.

### Value

A list that contains the following items:

- `pape`: The estimated Population Average Prescription Effect.
- `sd`: The estimated standard deviation of PAPE.

### Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, [https://www.michaellz.com/](https://www.michaellz.com/);

### References

Imai and Li (2019). “Experimental Evaluation of Individualized Treatment Rules”,

---

**PAPE**  
*Estimation of the Population Average Prescription Effect in Randomized Experiments*
Examples

\[
\begin{align*}
T &= c(1,0,1,0,1,0,1,0) \\
\text{That} &= c(0,1,1,0,0,1,1,0) \\
Y &= c(4,5,0,2,4,1,-4,3) \\
papelist &\leftarrow \text{PAPE}(T, \text{That}, Y) \\
papelist$pape \\
papelist$sd
\end{align*}
\]

Description

This function estimates the Population Average Prescription Effect with and without a budget constraint. The details of the methods for this design are given in Imai and Li (2019).

Usage

\text{PAPEcv}(T, \text{That}, Y, \text{ind}, \text{budget} = \text{NA}, \text{centered} = \text{TRUE})

Arguments

- \text{T} \quad \text{A vector of the unit-level binary treatment receipt variable for each sample.}
- \text{That} \quad \text{A matrix where the } i\text{th column is the unit-level binary treatment that would have been assigned by the individualized treatment rule generated in the } i\text{th fold. If budget is specified, please ensure that the percentage of treatment units of That is lower than the budget constraint.}
- \text{Y} \quad \text{The outcome variable of interest.}
- \text{ind} \quad \text{A vector of integers (between 1 and number of folds inclusive) indicating which testing set does each sample belong to.}
- \text{budget} \quad \text{The maximum percentage of population that can be treated under the budget constraint. Should be a decimal between 0 and 1. Default is NA which assumes no budget constraint.}
- \text{centered} \quad \text{If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.}

Value

A list that contains the following items:

- \text{pape} \quad \text{The estimated Population Average Prescription Effect.}
- \text{sd} \quad \text{The estimated standard deviation of PAPE.}
Author(s)
Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;

References

Examples
T = c(1,0,1,0,1,0,1,0)
That = matrix(c(0,1,1,0,0,1,1,0,1,0,0,1,1,0,0,1), nrow = 8, ncol = 2)
Y = c(4,5,0,2,4,1,-4,3)
ind = c(rep(1,4),rep(2,4))
papelist <- PAVEcv(T, That, Y, ind)
papelist$pape
papelist$sd

---

**PAV**

Estimation of the Population Average Value in Randomized Experiments

Description
This function estimates the Population Average Value. The details of the methods for this design are given in Imai and Li (2019).

Usage

```
PAV(T, That, Y, centered = TRUE)
```

Arguments

- **T**: A vector of the unit-level binary treatment receipt variable for each sample.
- **That**: A vector of the unit-level binary treatment that would have been assigned by the individualized treatment rule. If budget is specified, please ensure that the percentage of treatment units of That is lower than the budget constraint.
- **Y**: A vector of the outcome variable of interest for each sample.
- **centered**: If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.

Value

A list that contains the following items:

- **pav**: The estimated Population Average Value.
- **sd**: The estimated standard deviation of PAV.
Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>, https://www.michaellz.com/;

References

Imai and Li (2019). “Experimental Evaluation of Individualized Treatment Rules”,

Examples

```r
T = c(1, 0, 1, 0, 1, 0, 1, 0)
That = c(0, 1, 1, 0, 0, 1, 1, 0)
Y = c(4, 5, 0, 2, 4, 1, -4, 3)
pavlist <- PAV(T, That, Y)
pavlist$pav
pavlist$sd
```

Description

This function estimates the Population Average Value. The details of the methods for this design are given in Imai and Li (2019).

Usage

```
PAVcv(T, That, Y, ind, centered = TRUE)
```

Arguments

- `T` A vector of the unit-level binary treatment receipt variable for each sample.
- `That` A matrix where the i-th column is the unit-level binary treatment that would have been assigned by the individualized treatment rule generated in the i-th fold. If budget is specified, please ensure that the percentage of treatment units of That is lower than the budget constraint.
- `Y` The outcome variable of interest.
- `ind` A vector of integers (between 1 and number of folds inclusive) indicating which testing set does each sample belong to.
- `centered` If TRUE, the outcome variables would be centered before processing. This minimizes the variance of the estimator. Default is TRUE.

Value

A list that contains the following items:

- `pav` The estimated Population Average Value.
- `sd` The estimated standard deviation of PAV.
Author(s)

Michael Lingzhi Li, Technology and Operations Management, Harvard Business School <mili@hbs.edu>,
https://www.michaellz.com/;

References

Imai and Li (2019). “Experimental Evaluation of Individualized Treatment Rules”,

Examples

```r
T = c(1,0,1,0,1,0,1,0)
That = matrix(c(0,1,1,0,0,1,1,0,1,0,0,1,1,0,0,1), nrow = 8, ncol = 2)
Y = c(4,5,0,2,4,1,-4,3)
ind = c(rep(1,4),rep(2,4))
pavlist <- PAVcv(T, That, Y, ind)
pavlist$pav
pavlist$sd
```

---

`plot.itr` *Plot the AUPEC curve*

### Description

Plot the AUPEC curve

### Usage

```r
## S3 method for class 'itr'
plot(x, ...)
```

### Arguments

- `x` An object of `evaluate_itr()` class. This is typically an output of `evaluate_itr()` function.
- `...` Further arguments passed to the function.

### Value

A plot of ggplot2 object.
### plot_estimate

**Plot the GATE estimate**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plot the GATE estimate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>plot_estimate(x, type, ...)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
</tr>
<tr>
<td>type</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A plot of ggplot2 object.</td>
</tr>
</tbody>
</table>

### print.summary.itr

**Print**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usage</th>
</tr>
</thead>
</table>
| ## S3 method for class 'summary.itr'
| print(x, ...) |

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
print.summary.test_itr

Print

Description

Print

Usage

## S3 method for class 'summary.test_itr'
print(x, ...)

Arguments

x   An object of summary.test_itr class. This is typically an output of summary.test_itr() function.
...
  Other parameters.

star  Tennessee’s Student/Teacher Achievement Ratio (STAR) project

Description

A longitudinal study experimentally evaluating the impacts of class size in early education on various outcomes (Mosteller, 1995)

Usage

star

Format

A data frame with 1911 observations and 14 variables:

- **treatment** A binary treatment indicating whether a student is assigned to small class and regular class without an aid
- **g3tlangss** A continuous variable measuring student’s writing scores
- **g3treadss** A continuous variable measuring student’s reading scores
- **g3tmathss** A continuous variable measuring student’s math scores
- **gender** Students’ gender
- **race** Students’ race
- **birthmonth** Students’ birth month
- **birthyear** Students’ birth year
**SCHLURBN** Urban or rural  
**GKENRMNT** Enrollment size  
**GRDRANGE** Grade range  
**GKFRLNCH** Number of students on free lunch  
**GKBUSED** Number of students on school buses  
**GKWHITE** Percentage of white students

---

**summary.itr**  
*Summarize estimate_itr output*

### Description
Summarize estimate_itr output

### Usage
```r
## S3 method for class 'itr'
summary(object, ...)
```

### Arguments
- **object**  
  An object of estimate_itr class (typically an output of estimate_itr() function).
- **...**  
  Other parameters.

---

**summary.test_itr**  
*Summarize test_itr output*

### Description
Summarize test_itr output

### Usage
```r
## S3 method for class 'test_itr'
summary(object, ...)
```

### Arguments
- **object**  
  An object of test_itr class (typically an output of test_itr() function).
- **...**  
  Other parameters.
test_itr Conduct hypothesis tests

Description

Conduct hypothesis tests

Usage

test_itr(fit, nsim = 1000, ...)

Arguments

fit Fitted model. Usually an output from estimate_itr
nsim Number of Monte Carlo simulations used to simulate the null distributions. Default is 1000.
... Further arguments passed to the function.

Value

An object of test_itr class
Index

* datasets
  star, 28

* evaluation
  AUPEC, 3
  AUPECcv, 4
  consist.test, 6
  consistcv.test, 7
  GATE, 15
  GATEcv, 16
  het.test, 17
  hetcv.test, 18
  PAPD, 19
  PAPDcv, 20
  PAPE, 22
  PAPEcv, 23
  PAV, 24
  PAVcv, 25

AUPEC, 3
AUPECcv, 4

compute_qoi, 5
compute_qoi_user, 5
consistent.test, 6
consistentcv.test, 7
create_ml_args, 8
create_ml_args_bart, 9
create_ml_args_bartc, 9
create_ml_args_causalforest, 9
create_ml_args_lasso, 10
create_ml_args_superLearner, 10
create_ml_args_svm, 10
create_ml_args_svm_cls, 11
create_ml_arguments, 11

estimate_itr, 12
evaluate_itr, 13

fit_itr, 14

GATE, 15