Package 'debiasedTrialEmulation'

March 20, 2026

Type Package

Title Pipeline for Debiased Target Trial Emulation

Version 0.1.2

Description Supports propensity score-based methods—including matching, stratification, and weighting—for estimating causal treatment effects. It also implements calibration using negative control outcomes to enhance robustness. 'debiasedTrialEmulation' facilitates effect estimation for both binary and time-to-event outcomes, supporting risk ratio (RR), odds ratio (OR), and hazard ratio (HR) as effect measures. It integrates statistical modeling and visualization tools to assess covariate balance, equipoise, and bias calibration. Additional methods—including approaches to address immortal time bias, information bias, selection bias, and informative censoring—are under development. Users interested in these extended features are encouraged to contact the package authors.

Imports
dplyr,janitor,cobalt,MatchIt,geex,glmnet,survival,ggplot2,ParallelLogger,EmpiricalCalibration,purrr

Depends R (>= 3.5.0)

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Bingyu Zhang [aut, cre],
Yiwen Lu [aut],
Huilin Tang [aut],
Dazheng Zhang [aut],
Yuqing Lei [aut],
Tingyin Wang [aut],
Siqi Chen [aut],
Yong Chen [aut]

Maintainer Bingyu Zhang <bingyuz7@sas.upenn.edu>

Repository CRAN

Date/Publication 2026-03-20 00:30:03 UTC

Contents

`calibrate_TTE` *Negative Control Calibration for Target Trial Emulation*

Description

Performs calibration with negative control outcomes to further reduce confounding bias in Target Trial Emulation results.

Usage

```
calibrate_TTE(tte_obj = NULL, custom_results = NULL, custom_nco_results = NULL)
```

Arguments

`tte_obj`	Optional. An object of class "TTE" from the TTE_pipeline function. If provided, the function will use the negative control results from this object. Either this or both custom_results and custom_nco_results must be provided.
`custom_results`	Optional. A data frame containing the primary outcome results if no TTE object is available. Must contain columns 'names_outcome', 'logEst', and 'seLogEst'.
`custom_nco_results`	Optional. A data frame containing the negative control outcome results if no TTE object is available. Must contain columns 'names_outcome', 'logEst', and 'seLogEst'.

Value

An object of class "dTTE" containing both the TTE results and calibration results

Examples

```
library("dplyr")
data(demo_data)
# First run TTE pipeline
xvars <- c("eth_cat", "age_cat", "sex", "cohort_entry_month",
           "obese", "pmca_index", "n_ed", "n_inpatient",
           "n_tests", "imm_date_diff_grp", "medical_1",
           "medical_2", "medical_3", "medical_4", "medical_5")
yvars1 <- colnames(demo_data %>% select(starts_with("visits_")))
ncovars1 <- colnames(demo_data %>% select(starts_with("nco_visits_")))
tte_result <- TTE_pipeline(demo_data, xvars=xvars, yvars=yvars1, ncovars=ncovars1,
                           ps_type="Matching", outcome_measure="RR")

# Then calibrate results
dtte_result <- calibrate_TTE(tte_obj = tte_result)

# Alternatively, provide custom results
df_results <- data.frame(
  names_outcome = c("outcome1", "outcome2"),
  logEst = c(0.1, -0.2),
  seLogEst = c(0.05, 0.08)
)
df_nco_results <- data.frame(
  names_outcome = c("nco1", "nco2", "nco3"),
  logEst = c(0.02, -0.03, 0.01),
  seLogEst = c(0.03, 0.04, 0.02)
)
dtte_result <- calibrate_TTE(custom_results = df_results, custom_nco_results = df_nco_results)
```

---

computePreferenceScore

Compute Preference Score

Description

Computes preference scores based on the propensity scores.

Usage

```
computePreferenceScore(data, unfilteredData = NULL)
```

Arguments

data	A data frame containing propensity scores.
unfilteredData	Optional dataset to compute proportions from.

Value

A data frame with added preference scores.

`computeWeights` *Compute Propensity Score Weights*

Description

Computes inverse probability treatment weights (IPTW) for ATE or ATT estimation.

Usage

```
computeWeights(population, estimator = "ate")
```

Arguments

`population`	A data frame containing treatment assignments and propensity scores.
`estimator`	Type of estimator, either "ate" (average treatment effect) or "att" (average treatment effect on the treated).

Value

A vector of computed weights.

`Compute_weight` *Compute Weights for Stratification*

Description

Computes inverse probability weights for stratification-based adjustment.

Usage

```
Compute_weight(data)
```

Arguments

`data`	A data frame containing strata IDs and treatment assignments.

Value

A data frame with row IDs and computed weights.

demo_data *Example Dataset for Debiased Trial Emulation*

Description

'demo_data' is a simulated dataset used to demonstrate the functionality of the 'debiasedTrialEmulation' package. It includes patient demographic information, treatment assignment, covariates, clinical outcomes, and negative control outcomes for evaluating treatment effects using propensity score methods.

The dataset contains 50,000 observations and 93 variables, including:

- **Demographic variables**: Ethnicity, age, sex, and cohort entry month. - **Treatment assignment**: Binary treatment indicator. - **Covariates**: Baseline health conditions and healthcare utilization variables. - **Primary outcomes**: Binary and time-to-event outcomes related to cardiovascular health. - **Negative control outcomes (NCOs)**: Outcomes used for bias calibration.

Usage

```
data(demo_data)
```

Format

A data frame with 50,000 rows and 93 variables

estEffect_matching *Estimate Treatment Effects using Propensity Score Matching*

Description

Computes effect estimates using propensity score matching to reduce confounding.

Computes effect estimates using propensity score stratification to adjust for confounding.

Computes effect estimates using propensity score weighting to balance covariates between treatment groups.

Usage

```
estEffect_matching(form, data, yvars, ncovars, distance, outcome_measure)

estEffect_stratification(form, data, yvars, ncovars, distance, outcome_measure)

estEffect_weighting(form, data, yvars, ncovars, distance, outcome_measure)
```

Arguments

form	A formula specifying the treatment assignment model.
data	A dataset containing covariates and treatment assignment.
yvars	A character vector of outcome variable names.
ncovars	A character vector of negative control outcome variable names.
distance	The method for estimating propensity scores ("glm").
outcome_measure	The outcome measure to estimate: "RR" (Risk Ratio), "OR" (Odds Ratio), or "HR" (Hazard Ratio).

Value

List of components

List of components

List of components

GetSMD *Compute Standardized Mean Differences (SMD)*

Description

Computes the standardized mean differences for covariates before and after adjustment.

Usage

```
GetSMD(data, treat, weights = NULL, std = TRUE)
```

Arguments

data	A data frame containing covariates.
treat	A binary variable indicating treatment assignment.
weights	Optional weight vector.
std	Logical; whether to standardize.

Value

A data frame with covariate names and their standardized mean differences.

`get_OR_matching` *Estimate Odds Ratio (OR) after Propensity Score Matching*

Description

Computes the odds ratio for a binary outcome after applying propensity score matching.

Computes the risk ratio for a binary outcome after applying propensity score matching.

Computes the hazard ratio for a time-to-event outcome after propensity score matching.

Computes OR, RR, and HR for a binary or time-to-event outcome after stratification.

Computes the risk ratio for a binary outcome after applying propensity score stratification.

Computes the hazard ratio for a time-to-event outcome after propensity score stratification.

Computes the odds ratio for a binary outcome after applying propensity score weighting.

Computes the risk ratio for a binary outcome after applying propensity score weighting.

Computes the hazard ratio for a time-to-event outcome after applying propensity score weighting.

Usage

```
get_OR_matching(data, names_outcome)

get_RR_matching(data, names_outcome)

get_HR_matching(data, names_outcome)

get_OR_stratification(data, names_outcome)

get_RR_stratification(data, names_outcome)

get_HR_stratification(data, names_outcome)

get_OR_weighting(data, names_outcome, IPTW)

get_RR_weighting(data, names_outcome, IPTW)

get_HR_weighting(data, names_outcome, IPTW)
```

Arguments

`data`	A dataset containing treatment assignment and outcome variables.
`names_outcome`	A character vector of outcome variable names.
`IPTW`	A numeric vector of inverse probability of treatment weights.

Value

A data frame with the estimated log odds ratio, standard error, and p-value.

plot.dTTE *Plot Method for dTTE Objects (Calibration Only)*

Description

Plots only the calibration graph for the dTTE pipeline output.

Usage

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

Arguments

x	An object of class "dTTE".
...	Additional arguments passed to plotting functions.

plot.TTE *Plot Method for TTE Objects*

Description

Plots diagnostic graphs for the TTE pipeline output.

Usage

```
## S3 method for class 'TTE'
plot(x, which = c("SMD", "Equipoise"), ...)
```

Arguments

x	An object of class "TTE".
which	A character vector specifying which plot(s) to display. Options are "SMD" (Standardized Mean Differences) and "Equipoise" (Equipoise plot). Default shows all plots.
...	Additional arguments passed to plotting functions.

plot_SMD_matching *Plot Standardized Mean Differences (SMD) for Matching*

Description

Generates a plot of standardized mean differences before and after propensity score matching.

Generates a plot showing the distribution of preference scores to assess equipoise after matching.

Generates an SMD plot to compare balance before and after stratification.

Generates a plot showing the distribution of preference scores to assess equipoise after stratification.

Generates an SMD plot to compare balance before and after propensity score weighting.

Generates a plot showing the distribution of preference scores to assess equipoise after weighting.

Creates a plot of propensity score distributions using density or histogram visualization.

Creates a plot showing the top covariates with the largest standardized mean differences before and after matching.

Usage

```
plot_SMD_matching(m.out)

plot_Equipoise_matching(data, m.out)

plot_SMD_stratification(stratifiedPop, xvars)

plot_Equipoise_stratification(data)

plot_SMD_weighting(data)

plot_Equipoise_weighting(data)

plotPs(
  data,
  unfilteredData = NULL,
  scale = "preference",
  type = "density",
  binWidth = 0.05,
  targetLabel = "Target",
  comparatorLabel = "Comparator",
  showCountsLabel = FALSE,
  showAucLabel = FALSE,
  showEquiposeLabel = FALSE,
  equipoiseBounds = c(0.3, 0.7),
  unitOfAnalysis = "subjects",
  title = NULL,
  fileName = NULL
```

```
)

plotCovariateBalanceOfTopVariables(
  balance,
  n = 20,
  maxNameWidth = 100,
  title = NULL,
  fileName = NULL,
  beforeLabel = "before matching",
  afterLabel = "after matching"
)
```

Arguments

m.out	The output from 'MatchIt', containing matched data.
data	A dataset containing treatment and propensity scores.
stratifiedPop	The dataset containing stratified propensity scores.
xvars	The covariate names to assess balance.
scale	The scale to use: "preference" or "propensity".
type	The type of plot: "density", "histogramCount", or "histogramProportion".
binWidth	The bin width for histograms (default = 0.05).
targetLabel	Label for the treated group.
comparatorLabel	Label for the control group.
showCountsLabel	Logical; whether to show sample counts.
showAucLabel	Logical; whether to show AUC.
showEquiposeLabel	Logical; whether to indicate equipoise range.
equipoiseBounds	A numeric vector of two values defining the equipoise range (default = c(0.3, 0.7)).
unitOfAnalysis	Unit label for counts (e.g., "subjects").
title	Optional title for the plot.
fileName	Optional file name to save the plot.
balance	A data frame containing standardized mean differences.
n	Number of top covariates to display.
beforeLabel	Label for pre-matching imbalance.
afterLabel	Label for post-matching balance.
unfilteredData	A logical indicating whether to include unfiltered data in the plot.
maxNameWidth	An integer specifying the maximum width for variable names.

Value

A 'ggplot2' object showing balance improvement after matching.

print.dTTE *Print Method for dTTE Objects*

Description

Prints a concise summary of the dTTE pipeline output.

Usage

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

Arguments

x	An object of class "dTTE".
...	Additional arguments (currently ignored).

print.TTE *Print Method for TTE Objects*

Description

Prints a concise summary of the TTE pipeline output.

Usage

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

Arguments

x	An object of class "TTE".
...	Additional arguments (currently ignored).

stratifyByPs *Stratify Population by Propensity Score*

Description

Assigns individuals to strata based on their propensity scores.

Usage

```
stratifyByPs(
  population,
  numberOfStrata = 5,
  stratificationColumns = c(),
  baseSelection = "all"
)
```

Arguments

population	A data frame containing row IDs, treatment assignments, and propensity scores.
numberOfStrata	Number of strata to create.
stratificationColumns	Additional columns to use for stratification.
baseSelection	Defines which group is used to determine strata cutoffs ("all", "target", or "comparator").

Value

A data frame with stratum assignments.

summary.dTTE *Summary Method for dTTE Objects*

Description

Provides a detailed summary of the dTTE pipeline output.

Usage

```
## S3 method for class 'dTTE'
summary(x, ...)
```

Arguments

x	An object of class "dTTE".
...	Additional arguments (currently ignored).

summary.TTE *Summary Method for TTE Objects*

Description

Provides a detailed summary of the TTE pipeline output.

Usage

```
## S3 method for class 'TTE'
summary(x, ...)
```

Arguments

x	An object of class "TTE".
...	Additional arguments (currently ignored).

trimByPsQuantile *Trim Propensity Scores*

Description

Trims propensity scores by removing extreme values at both ends of the distribution.

Usage

```
trimByPsQuantile(propensityScore, trimFraction = 0.05)
```

Arguments

propensityScore	A numeric vector of propensity scores.
trimFraction	Fraction of extreme values to trim (default 0.05).

Value

A vector of indices indicating which scores to keep.

TTE_pipeline *Target Trial Emulation (TTE) Pipeline*

Description

Implements a Target Trial Emulation pipeline using propensity score methods, including matching, weighting, and stratification.

Usage

```
TTE_pipeline(data, xvars, yvars, ncovars = NULL, ps_type, outcome_measure)
```

Arguments

data	A dataset containing treatment assignment, covariates, and outcomes.
xvars	A character vector of covariate names used for propensity score estimation.
yvars	A character vector of primary outcome variable names.
ncovars	Optional. A character vector of negative control outcome variable names.
ps_type	The propensity score method: "Matching", "Stratification", or "Weighting".
outcome_measure	The outcome measure to estimate: "RR" (Risk Ratio), "OR" (Odds Ratio), or "HR" (Hazard Ratio).

Value

An object of class "TTE" containing the propensity score analysis results

Examples

```
library("dplyr")
data(demo_data)
xvars <- c("eth_cat", "age_cat", "sex", "cohort_entry_month",
           "obese", "pmca_index", "n_ed", "n_inpatient",
           "n_tests", "imm_date_diff_grp", "medical_1",
           "medical_2", "medical_3", "medical_4", "medical_5")
yvars1 <- colnames(demo_data %>% select(starts_with("visits_")))
yvars2 <- colnames(demo_data %>% select(starts_with("event_")))
# without negative controls
TTE_pipeline(demo_data, xvars=xvars, yvars=yvars1,
            ps_type="Matching", outcome_measure="RR")
# with negative controls
ncovars1 <- colnames(demo_data %>% select(starts_with("nco_visits_")))
TTE_pipeline(demo_data, xvars=xvars, yvars=yvars1,
             ncovars=ncovars1, ps_type="Matching", outcome_measure="RR")
```

Index