

Package ‘RCT’

July 13, 2020

Title Assign Treatments, Power Calculations, Balances, Impact
Evaluation of Experiments

Version 1.0.3

Description Assists in the whole process of designing and evaluating Randomized Control Trials.
Robust treatment assignment by strata/blocks, that handles misfits;
Power calculations of the minimum detectable treatment effect
Balance tables of T-test of covariates;
Balance Regression: (treatment ~ all x variables) with F-test of null model;
Impact_evaluation: Impact evaluation regressions. This function
gives you the option to include control_vars, fixed effect variables,
cluster variables (for robust SE), multiple endogenous variables and
multiple heterogeneous variables (to test treatment effect heterogeneity)
summary_statistics: Function that creates a summary statistics table with statistics
rank_observations_in_n_groups: Creates a factor variable with n groups. Each group has
a min and max label attach to each category.
Athey, Susan, and Guido W. Imbens (2017) <arXiv:1607.00698>.

License GPL-2

Encoding UTF-8

LazyData true

Imports dplyr, purrr, glue, rlang, tidyr, stringr, MASS, pracma, lfe,
broom, forcats, magrittr, ggplot2, utils, tidyselect (>= 1.0.0)

Suggests knitr, rmarkdown, testthat, qpdf

RoxygenNote 7.1.0

VignetteBuilder knitr

NeedsCompilation no

Author Isidoro Garcia-Urquieta [aut, cre]

Maintainer Isidoro Garcia-Urquieta <isidoro.gu@gmail.com>

Repository CRAN

Date/Publication 2020-07-13 05:10:02 UTC

R topics documented:

balance_regression	2
balance_table	3
impact_eval	3
ntile_label	5
RCT	6
summary_statistics	7
tau_min	7
tau_min_probability	9
treatment_assign	10
Index	12

balance_regression	<i>balance_regression()</i> Runs a LPM of treatment status against all covariates (treatment~X'B).
--------------------	--

Description

balance_regression() Runs a LPM of treatment status against all covariates (treatment~X'B).

Usage

```
balance_regression(data, treatment)
```

Arguments

data	A data.frame, tibble or data.table
treatment	a string with treatment status column

Details

This functions runs a Linear Probability model of each treatment group & control on all the columns in data. For instance, if treatment column has values of (0,1,2), balance_regression will run two models: 1) LPM(treatment(0,1)~X'b) and 2) LPM(treatment(0,2)~X'b). The value are the regression tables and details of the F_test of these models.

Value

A list: regression_tables = regression output of treatment against all covariates, F_test = table with the F tests of each regression

Examples

```
data <- data.frame(x1 = rnorm(n = 100, mean = 100, sd = 15), x2 = rnorm(n = 100, mean = 65),
  treat = rep(c(0,1,2,3,4), each = 20))
balance_regression(data, treatment = "treat")
```

balance_table	<i>Creates balance table for the X variables across treatment status</i>
---------------	--

Description

Creates balance table for the X variables across treatment status

Usage

```
balance_table(data, treatment)
```

Arguments

data	A data.frame, tibble or data.table
treatment	a string with treatment status column

Details

balance_table() performs t.test(X~treatment) for each X column in data. Every value of treatment i.e 1,2,3,...N is compared against control value (0) or the first value of the treatment column. For instance, If treatment column has values of (0,1,2,3), balance_table will return: the mean value of each treatment (for all X's), and the p_values of the t.test of (1,2,3) against treatment = 0.

Value

A tibble with Mean_value of each treatment status and p_values

Examples

```
data <- data.frame(x1 = rnorm(n = 100, mean = 100, sd = 15),
                  x2 = rnorm(n = 100, mean = 65),
                  treatment = rep(c(0,1,2,3,4), each = 20))
balance_table(data, "treatment")
```

impact_eval	<i>Impact Evaluation of Treatment Effects</i>
-------------	---

Description

Impact Evaluation of Treatment Effects

Usage

```

impact_eval(
  data,
  endogenous_vars,
  treatment,
  heterogenous_vars,
  cluster_vars = "0",
  fixed_effect_vars = "0",
  control_vars
)

```

Arguments

<code>data</code>	A <code>data.frame</code> , <code>tibble</code> or <code>data.table</code>
<code>endogenous_vars</code>	Vector of Y's on which treatment effects will be evaluated
<code>treatment</code>	Variable indicating the treatment status
<code>heterogenous_vars</code>	Vector of variables for which you wish to assess treatment distributions/heterogeneities.
<code>cluster_vars</code>	Vector of variables to cluster the standard errors. Default is without clustered std errors
<code>fixed_effect_vars</code>	Vector of variables to add as fixed effects. Default is without fixed effects
<code>control_vars</code>	Vector of variables to control for in the evaluation. Default is without controls

Details

This function carries out the evaluation of treatment effects on endogenous variables. It automatically runs the regressions of all the `endogenous_vars` supplied & all the combinations of `endogenous_vars` and `heterogenous_vars`. Additionally, the function has the option of include `fixed_effects`, `contols` and `cluster variables` for clustered std errors.

Value

`impact_eval()` returns a list of regression tables. The names of the list are the same as the endogenous variables. for heterogeneities the names are `endogenous_var_heterogenous_var`

Examples

```

data <- data.frame(y_1 = rnorm(n = 100, mean = 100, sd = 15),
  y_2 = rnorm(n = 100, mean = 8, sd = 2),
  treat = rep(c(0,1,2,3), each = 25),
  heterogenous_var1 = rep(c("X_Q1", "X_Q2", "X_Q3", "X_Q4"), times = 25),
  cluster_var1 = rep(c(1:5), times = 20),
  fixed_effect_var1 = rep(c(1,2), times = 50),
  control_var1 = rnorm(n = 100, mean = 20, sd = 1))

evaluation<-impact_eval(data = data,

```

```

    endogenous_vars = c("y_1", "y_2"),
    treatment = "treat",
    heterogenous_vars = c("heterogenous_var1"),
    cluster_vars = "cluster_var1", fixed_effect_vars = c("fixed_effect_var1"),
    control_vars = c("control_var1")

```

ntile_label

ntile_label() ranks observations in n groups, with labels

Description

ntile_label() ranks observations in n groups, with labels

Usage

```
ntile_label(var, n, digits = 0)
```

Arguments

var	The variable wished to be ntile_label
n	rank the variable in n groups
digits	How many digits to include in the label

Details

n_tile_label is very similar to ntile from dplyr. But n_tile_label creates the n groups and then labels them. For each group i, the value of the ntile_label is [min(i) - max(i)].

Value

A ordered factor vector of each n group. The value has the form of [min(n_i) - max(n_i)]

Examples

```

data <- data.frame(y_1 = rbinom(n = 100, size = 1, prob = 0.3),
                  y_2 = rnorm(n = 100, mean = 8, sd = 2))
data$y_1_2 <- ntile_label(data$y_1, n = 2, digits = 0)
data$y_2_4 <- ntile_label(data$y_2, n = 4, digits = 1)

```

RCT	<i>Designing, random assigning and evaluating Randomized Control Trials</i>
-----	---

Description

RCT provides three important group of functions: a) functions for pre-processing the design of the RCT b) Functions for assigning treatment status and checking for balances c) Function for evaluating the impact of the RCT

Details

RCT helps you focus on the statistics of the randomized control trials, rather than the heavy programming lifting. RCT helps you in the whole process of designing and evaluating a RCT. 1. Clean and summarise the data in which you want to randomly assign treatment 2. Decide the share of observations that will go to control group 3. Decide which variables to use for strata building 4. Robust Random Assignment by strata/blocks 5 Impact evaluation of all y's and heterogeneities To learn more about RCT, start with the vignette: `browseVignettes(package = "RCT")`

RCT functions

`treatment_assign`: Robust treatment assign by strata/blocks

`impact_eval`: Automatized impact evaluation with heterogeneous treatment effects

`balance_table`: Balance tables for any length of covariates

`balance_regression`: LPM of treatment status against covariates with F-test

`tau_min`: Computation of the minimum detectable effect between control & treatment units

`tau_min_probability`: Computation of the minimum detectable effect between control & treatment units for dicotomical y-vars

`summary_statistics`: Summary statistics of all numeric columns in your data

`ntile_label`: Rank and divide observations in n groups, with label

Author(s)

Isidoro Garcia Urquieta, isidoro.gu@gmail.com

References

Athey, Susan, and Guido W. Imbens (2017) "The Econometrics Randomized Experiments". Handbook of economic field experiments. <https://arxiv.org/abs/1607.00698>

See Also

Useful links: <https://github.com/isidorogu/RCT> Report bugs at <https://github.com/isidorogu/RCT/issues>

summary_statistics	<i>summary_statistics()</i> Creates summary statistics table of all numeric variables in data
--------------------	---

Description

summary_statistics() Creates summary statistics table of all numeric variables in data

Usage

```
summary_statistics(
  data,
  probs = c(0, 0.05, 0.1, 0.25, 0.5, 0.75, 0.9, 0.95, 1),
  na.rm = T
)
```

Arguments

data	A data.frame, tibble or data.table
probs	The quantiles to compute. Default is c(0, 0.05, 0.1, 0.25, 0.5, 0.75, 0.9, 0.95, 1)
na.rm	whether to exclude NA's from calculations

Details

This function computes the selected quantiles, mean and N values of all the numeric columns of data.

Value

A tibble with the Mean, N (not NA) and probs selects for each numeric column

Examples

```
data <- data.frame(x = c(1:5), y = c(100, 200, 300, 410, 540), z = rep("c", 5))
summary_statistics(data)
```

tau_min	<i>tau_min()</i> computes the minimum detectable difference between control group and each treatment
---------	--

Description

tau_min() computes the minimum detectable difference between control group and each treatment

Usage

```
tau_min(
  outcome_var,
  N,
  power = 0.8,
  significance = 0.05,
  share_control,
  n_groups = 2
)
```

Arguments

outcome_var	the variable for which you wish to test the impact of treatment
N	number of observations in the RCT, usually <code>nrow(data)</code>
power	The level of power of the test ($1 - \Pr(\text{Reject } H_0 \mid H_0 \text{ True})$). Default is 0.8
significance	The level of significance of the test $\Pr(\text{Reject } H_0 \mid H_0 \text{ False})$. Default is 0.05
share_control	The share of observations in N assigned to control. This argument allows for sequences (i.e. <code>seq(0,1,0.1)</code>)
n_groups	Number of groups (control + # treatment groups)

Details

This function calculates the minimum difference that could show significant $E[Y(1)-Y(0)] = \tau$, between any two given groups (e.g. control vs each treatment), given the population size (N), the outcome variable, power and significance

Value

A tibble with the share_control and N observations in control group (N_control), the share and N of each treatment `c(share_ti, N_ti)`, total share of treatment rows and N treated (share_treat, N_treat), N, the minimum detectable difference between control and all treatments together (tau_min_global), the minimum detectable difference between control and each treatment (tau_min_each_treat)

Examples

```
data <- data.frame(y_1 = rbinom(n = 100, size = 1, prob = 0.3),
                  y_2 = rnorm(n = 100, mean = 8, sd = 2))
tau_min(data$y_1, N = nrow(data), share_control = seq(0,1,0.1), n_groups = 3)
```

tau_min_probability *tau_min_probability()* computes the minimum detectable difference between control group and each treatment for a dicotomical variable

Description

tau_min_probability() computes the minimum detectable difference between control group and each treatment for a dicotomical variable

Usage

```
tau_min_probability(
  prior,
  N,
  power = 0.8,
  significance = 0.05,
  share_control,
  n_groups = 2
)
```

Arguments

prior	Pr(Y=1).
N	number of observations in the RCT, usually nrow(data)
power	The level of power of the test (1 - Pr(Reject H ₀ H ₀ True)). Default is 0.8
significance	The level of significance of the test Pr(Reject H ₀ H ₀ False). Default is 0.05
share_control	The share of observations in N assigned to control. This argument allows for sequences (i.e. seq(0,1,0.1))
n_groups	Number of groups (control + # treatment groups)

Details

This function calculates the minimum difference that could show significant $\Pr[Y(1)-Y(0)] = \tau$, between any two given groups (e.g. control vs each treatment), given the population size (N), the outcome variable, power and significance

Value

A tibble with the share_control and N observations in control group (N_control), the share and N of each treatment c(share_ti, N_ti), total share of treatment rows and N treated (share_treat, N_treat), N, the minimum detectable difference between control and all treatments together (tau_min_global), the minimum detectable difference between control and each treatment (tau_min_each_treat)

Examples

```
tau_min_probability(0.4, N = 1000, share_control = seq(0,1,0.1), n_groups = 3)
```

treatment_assign	<i>treatment_assign() carries out robust treatment assignment by strata/blocks</i>
------------------	--

Description

treatment_assign() carries out robust treatment assignment by strata/blocks

Usage

```
treatment_assign(
  data,
  share_control,
  n_t = 2,
  strata_varlist,
  missfits = c("global", "NA", "strata"),
  seed = 1990,
  share_ti = rep(1/n_t - share_control/n_t, times = n_t),
  key
)
```

Arguments

data	A data.frame, tibble or data.table
share_control	share of the observations assigned to control group
n_t	Number of treatments groups
strata_varlist	vector of categorical variables to form the strata/blocks for random assignment. Should be in the form of vars(var1, var2, ...)
missfits	How to handle the misfits. Default is "global". See Carril (2016) for details.
seed	A number used to set.seed().
share_ti	The share of each treatment group. If NULL (Default), each treatment group will have equal share.
key	The key identifier column of data.

Details

This function creates a variable that indicates the treatment status. The random assignment is made by strata/blocks. It can handle equal or unequal treatment shares. Finally, it has three methods available to handle misfits (same as randtreat in STATA): "global": assigning the observations that couldn't be randomly assigned globally, "strata": assigning the observations that couldn't be randomly assigned by strata, "NA": set the the treat observations that couldn't be randomly assigned to NA.

Value

A list: "data" = the data with key, treat, strata, misfit column., "summary_strata" = A summary tibble with the membership of each strata and its size.

Examples

```
data<-data.frame(key = c(1:1000),
  ing_quartile = rep(c("Q1", "Q2", "Q3", "Q4"), each = 250),
  age_quartile = rep(c("Q1", "Q2", "Q3", "Q4"), times = 250))
assignment<-treatment_assign(data = data, share_control = 0.1, n_t = 3,
  strata_varlist = dplyr::vars(ing_quartile,
  age_quartile), missfits = "strata",
  seed = 1990, key = "key")
table(data$treat, useNA = "ifany")
prop.table(table(data$treat, useNA = "ifany"))
```

Index

balance_regression, [2](#)

balance_table, [3](#)

impact_eval, [3](#)

ntile_label, [5](#)

RCT, [6](#)

summary_statistics, [7](#)

tau_min, [7](#)

tau_min_probability, [9](#)

treatment_assign, [10](#)