Package ‘sensitivityCalibration’

October 14, 2022

Type Package

Title A Calibrated Sensitivity Analysis for Matched Observational Studies

Version 0.0.1

Author Bo Zhang

Maintainer Bo Zhang <bozhan@wharton.upenn.edu>

Description Implements the calibrated sensitivity analysis approach for matched observational studies. Our sensitivity analysis framework views matched sets as drawn from a superpopulation. The unmeasured confounder is modeled as a random variable. We combine matching and model-based covariate-adjustment methods to estimate the treatment effect. The hypothesized unmeasured confounder enters the picture as a missing covariate. We adopt a state-of-art Expectation Maximization (EM) algorithm to handle this missing covariate problem in generalized linear models (GLMs). As our method also estimates the effect of each observed covariate on the outcome and treatment assignment, we are able to calibrate the unmeasured confounder to observed covariates.


License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports ggplot2, relaimpo, splitstackshape, ggrepel, stringi, plotly

NeedsCompilation no

Repository CRAN

Date/Publication 2018-12-18 11:20:06 UTC

R topics documented:

  calibrate_anim .................................................. 2
  calibrate_one .................................................. 3
  CI_block_boot .................................................. 4
  EM_Algorithm .................................................. 5
  find_border .................................................... 6
  find_delta ..................................................... 7
calibrate_Anim

Make the dynamic calibration plot.

Description

This is another main function in the package. For a given p and the border of the sensitivity parameters (lambda, delta), a calibration plot is made for each (lambda, delta) pair on the border.

Usage

calibrate_Anim(border, q, u, p, degree, xmax, ymax, data_matched)

Arguments

- border: Border or frontier of the sensitivity parameters for a fixed p.
- q: Number of matched covariates plus treatment.
- u: Unmeasured confounder; u = c(1,0) if the unmeasured confounder is assumed to be binary.
- p: The probability vector corresponding to u; p = c(0.5, 0.5) if the unmeasured confounder is assumed to be Bernoulli(0.5).
- degree: Degree of freedom of the spline fit for the boundary.
- xmax: Maximum xlim of the plot.
- ymax: Maximum ylim of the plot.
- data_matched: The matched dataset.

Details

border is the dataframe returned by the function find_border. It has to contain at least (k+1) different lambda/delta pairs in order to fit a smoothing spline with k dfs.

Examples

data(NHANES_blood_lead_small_matched)
attach(NHANES_blood_lead_small_matched)

# Prepare the border
lambda_vec = c(seq(0.1,1.9,0.1), 2.2, 2.5, 3, 3.5, 4)
daleta_vec = c(7.31, 5.34, 4.38, 3.76, 3.18, 2.87, 2.55, 2.36, 2.16, 1.99, 1.86,
1.74, 1.63, 1.54, 1.44, 1.40, 1.31, 1.28, 1.22, 1.08, 0.964, 0.877, 0.815, 0.750)
border = data.frame(lambda_vec, delta_vec)
calibrate_one

Make the calibration plot.

Description
This is the main function in the package. Given a matched dataset and one particular (p, lambda, delta) triple, obtain corresponding coefficients of observed coefficients and plot them with the legend added. This graph is meant to provide an intuitive interpretation of the magnitude of the sensitivity parameters lambda and delta by contrasting them with the estimated coefficients of the observed covariates.

Usage

calibrate_one(lambda_vec, delta_vec, q, u, p, lambda, delta, label_vec, data_matched)

Arguments

lambda_vec
A vector of lambdas that define the border.

delta_vec
A vector of deltas that define the border.

q
Number of matched covariates plus treatment.

u
Unmeasured confounder; u = c(1, 0) if the unmeasured confounder is assumed to be binary.

p
The probability vector corresponding to u; p = c(0.5, 0.5) if the unmeasured confounder is assumed to be Bernoulli(0.5).

lambda
Sensitivity parameter that controls association between U and treatment assignment.

delta
Sensitivity parameter that controls association between U and response.

label_vec
A vector of characters of length q-1 consists of the names of observed/matched covariates.

data_matched
The matched dataset.

Details
border is the dataframe returned by the function find_border. It has to contain at least 7 different lambda/delta pairs in order to fit a smoothing spline with 6 dfs.

lambda and delta is a pair on the border.

label_vec is typically taken to be the columns names of the dataset, i.e., the names of the q-1 observed covariates.
CI_block_boot

Examples

```r
data(NHANES_blood_lead_small_matched)
attach(NHANES_blood_lead_small_matched)

# Prepare the lambda_vec and delta_vec
lambda_vec = c(seq(0.1, 1.9, 0.1), 2.2, 2.5, 3, 3.5, 4)
delta_vec = c(7.31, 5.34, 4.38, 3.76, 3.18, 2.87, 2.55, 2.36, 2.16, 1.99, 1.86,
1.74, 1.63, 1.54, 1.44, 1.40, 1.31, 1.28, 1.22, 1.08, 0.964, 0.877, 0.815, 0.750)

calibrate_one(lambda_vec, delta_vec, 9, c(1, 0), c(0.5, 0.5), 1, 0.492,
colnames(NHANES_blood_lead_small_matched)[1:8], NHANES_blood_lead_small_matched)

detach(NHANES_blood_lead_small_matched)
```

---

**CI_block_boot**

Construct the 95% confidence interval of the treatment effect given the
set of sensitivity parameters.

---

**Description**

This is the main function in the package. Given a dataset and sensitivity parameters (p, lambda, delta), the function returns 95% CI for the estimated treatment effect.

**Usage**

```
CI_block_boot(q, u, p, lambda, delta, data_matched, n_boot = 2000)
```

**Arguments**

- **q**: Number of matched covariates plus treatment.
- **u**: Unmeasured confounder; u = c(1, 0) if the unmeasured confounder is assumed to be binary.
- **p**: The probability vector corresponding to u; p = c(0.5, 0.5) if the unmeasured confounder is assumed to be Bernoulli(0.5).
- **lambda**: Sensitivity parameter that controls association between U and treatment assignment.
- **delta**: Sensitivity parameter that controls association between U and response.
- **data_matched**: The dataset after matching.
- **n_boot**: Number of bootstrap samples.
Details

If the number of matched covariates is k, then q = k + 1.
If the hypothesized unmeasured confounder is binary, then u = c(1,0) and p = c(p, 1-p).
data_matched should be in the following format: the first (q-1) columns are matched covariates, the
qth column is the treatment status, and the (q+1)th column is the response. See the NHANES_blood_lead_small_matched
dataset for an example.
Note the input for this function is a dataset before matching. To run this function, optmatch package
needs to be installed and loaded.

Examples

data(NHANES_blood_lead_small_matched)
attach(NHANES_blood_lead_small_matched)
CI_block_boot(9, c(1,0), c(0.5,0.5), 0, 0, NHANES_blood_lead_small_matched, n_boot = 10)
detach(NHANES_blood_lead_small_matched)

---

EM_Algorithm

Estimate the treatment effect for a matched dataset given the set of
sensitivity parameters.

Description

This is the main function in the package. Given a matched dataset and sensitivity parameters (p, lambda, delta), the function runs the EM algorithm by the method of weights and return estimated coefficients of the propensity score model and the outcome regression model.

Usage

EM_Algorithm(q, u, p, lambda, delta, data_matched, all_coef = FALSE,
            aug_data = FALSE, tol = 0.0001)

Arguments

q Number of matched covariates plus treatment.
u Unmeasured confounder; u = c(1,0) if the unmeasured confounder is assumed to be binary.
p The probability vector corresponding to u; p = c(0.5, 0.5) if the unmeasured confounder is assumed to be Bernoulli(0.5).
lambda Sensitivity parameter that controls association between U and treatment assignment.
delta Sensitivity parameter that controls association between U and response.
find_border

A matched dataset. See details below.

TRUE then all estimated coefficients are returned, FALSE then only the estimated treatment effect is returned.

TRUE then the augmented dataframe at the time of convergence is returned.

Tolerance for the algorithm convergence.

Details

If the number of matched covariates is k, then q = k + 1.

If the hypothesized unmeasured confounder is binary, then u = c(1,0) and p = c(p, 1-p).

data_matched should be in the following format: the first (q-1) columns are matched covariates, the qth column is the treatment status, the (q+1)th column is the column of unmeasured confounders U0, the (q+2)th column is the response, the last column, i.e., (q+3)th column, is the assignment of the matched set. We use the fullmatch function in the package optmatch to perform the full-matching. See NHANES_blood_lead_small_matched for an example of a matched dataset and the examples section therein for instructions on how to construct such a matched dataset.

Examples

data(NHANES_blood_lead_small_matched)
attach(NHANES_blood_lead_small_matched)

# Run the EM algorithm assuming no unmeasured confounding, i.e., lambda = delta = 0
EM_Algorithm(9, c(1,0), c(0.5,0.5), 0, 0, NHANES_blood_lead_small_matched)

# Run the EM algorithm assuming the magnitude of the unmeasured confounding is lambda = delta = 1
EM_Algorithm(9, c(1,0), c(0.5,0.5), 1, 1, NHANES_blood_lead_small_matched)

detach(NHANES_blood_lead_small_matched)

find_border

Find the lambda-delta boundary for a fixed sensitivity parameter p.

Description

Given the dataset, unmeasured confounder, sensitivity parameter p, and a sequence of lambda values, the function uses binary search to find a sequence of delta corresponding to each lambda in the lambda_vec such that the estimated 95% for the treatment effect barely covers 0. The function returns a dataframe consisting of lambda_vec and the corresponding deltas. See below for an example.

Usage

find_border(q, u, p, lambda_vec, start_value_low, start_value_high, data_matched, n_boot = 2000, tol = 0.01)
**$find_delta$**

**Arguments**

- **q**: Number of matched covariates plus treatment.
- **u**: Unmeasured confounder; $u = c(1,0)$ if the unmeasured confounder is assumed to be binary.
- **p**: The probability vector corresponding to $u$; $p = c(0.5, 0.5)$ if the unmeasured confounder is assumed to be Bernoulli(0.5).
- **lambda_vec**: A sequence of lambda values.
- **start_value_low**: Starting value for the binary search (the lower endpoint).
- **start_value_high**: Starting value for the binary search (the higher endpoint).
- **data_matched**: The dataset after matching.
- **n_boot**: Number of bootstrap samples used to approximate the CI.
- **tol**: Tolerance for the binary search.

**Details**

- `start_value_low` and `start_value_high` are user supplied numbers to start the binary search.

**Examples**

```r
data(NHANES_blood_lead_small_matched)
attach(NHANES_blood_lead_small_matched)

find_delta(9, c(1,0), c(0.5,0.5), c(0.5,1,1.5), 0, 4, NHANES_blood_lead_small_matched, n_boot = 1000)

detach(NHANES_blood_lead_small_matched)
```

**Description**

Estimate the maximum delta value for a given $p$ and $\lambda$, so that the estimated 95% confidence interval for the treatment effect is still significant. Note in order to run this function, optmatch package needs to be installed and loaded.

**Usage**

```r
find_delta(q, u, p, lambda, start_value_low, start_value_high, data_matched, n_boot = 200, tol = 0.01)
```
Arguments

q  Number of matched covariates plus treatment.
u  Unmeasured confounder; u = c(1,0) if the unmeasured confounder is assumed to be binary.
p  The probability vector corresponding to u; p = c(0.5, 0.5) if the unmeasured confounder is assumed to be Bernoulli(0.5).
lambda  A lambda value.
start_value_low  Starting value for the binary search (the lower endpoint).
start_value_high  Starting value for the binary search (the higher endpoint).
data_matched  The dataset after matching.
n_boot  Number of bootstrap samples used to approximate the CI.
tol  Tolerance for the binary search.

Details

start_value_low and start_value_high are user supplied numbers to start the binary search.

Examples

data(NHANES_blood_lead_small_matched)
attach(NHANES_blood_lead_small_matched)
find_delta(9, c(1,0), c(0.5,0.5), 1, 1, 3, NHANES_blood_lead_small_matched, n_boot = 1000)
detach(NHANES_blood_lead_small_matched)

NHANES_blood_lead  Second hand smoking and blood lead levels dataset from NHANES III.

Description

A dataset constructed from NHANES III.

Usage

data(NHANES_blood_lead)
**NHANES_blood_lead**

**Format**

A data frame with 4519 observations on the following 10 variables.

- **COT** treatment, 1 if cotinine level is between 0.563-14.9 ng/ml and 0 otherwise
- **DMARETHN** 1 if white, 0 if others
- **DMPPIR** Poverty income ratio
- **HFE1** 1 if the house is built before 1974, 0 if after 1974
- **HFE2** number of rooms in the house
- **HFHEUDCR** education level of the reference adult
- **HSAGEIR** age at the time of interview
- **HSFSIZER** size of the family
- **HSSEX** 1 if male, 0 if female
- **PBP** blood lead level

**Details**

We follow Mannino rt al. (2003) in constructing a dataset that includes children aged 4-16 years old for whom both serum cotinine levels and blood lead levels were measured in the Third National Health and Nutrition Examination Survey (NHANES III), along with the following variables: race/ethnicity, age, sex, poverty income ratio, education level of the reference adult, family size, number of rooms in the house, and year the house was constructed. The biomarker cotinine is a metabolite of nicotine and an indicator of second-hand smoke exposure. Treatment status is 1 if cotinine level is between 0.563-14.9 ng/ml and 0 otherwise. All continuous/ordinal variables are standardized by subtracting the mean and divided by 2 standard deviations so that they are more comparable to binary covariates (Gelman 2008).

**Source**

NHANES III, the Third US National Health and Nutrition Examination Survey.

**References**


**Examples**

data(NHANES_blood_lead)
NHANES_blood_lead_small

*A random subset of NHANES_blood_lead data.*

---

**Description**

A random subset of NHANES_blood_lead data for the purpose of testing.

**Usage**

```
data(NHANES_blood_lead_small)
```

**Format**

A random sample from the NHANES_blood_lead dataset. It consists of 500 instances and the same 10 variables as the NHANES_blood_lead data.

- **COP**  treatment, 1 if cotinine level is between 0.563-14.9 ng/ml and 0 otherwise
- **DMARETHN**  1 if white, 0 if others
- **DMPPIR**  Poverty income ratio
- **HFE1**  1 if the house is built before 1974, 0 if after 1974
- **HFE2**  number of rooms in the house
- **HFHEDUCR**  education level of the reference adult
- **HSAGEIR**  age at the time of interview
- **HSFSIZER**  size of the family
- **HSSEX**  1 if male, 0 if female
- **PBP**  blood lead level

**Details**

We take a 500 random sample from the NHANES_blood_lead dataset. This small dataset is primarily for the purpose of testing the algorithm.

**Source**

NHANES III, the Third US National Health and Nutrition Examination Survey.

**References**


**Examples**

```
data(NHANES_blood_lead_small)
```
Description

NHANES_blood_lead_small data after a full matching using the optmatch package

Usage

data(NHANES_blood_lead_small_matched)

Format

NHANES_blood_lead_small dataset after a full matching. It consists of 500 instances and the following 12 variables:

- COP  treatment, 1 if cotinine level is between 0.563-14.9 ng/ml and 0 otherwise
- DMARETHN 1 if white, 0 if others
- DMPPIR  Poverty income ratio
- HFE1  1 if the house is built before 1974, 0 if after 1974
- HFE2  number of rooms in the house
- HFHEDUCR  education level of the reference adult
- HSAGEIR  age at the time of interview
- HSFSIZER  size of the family
- HSSEX  1 if male, 0 if female
- PBP  blood lead level
- U0  placeholder for the hypothesized unmeasured confounder U
- matches  matched set assignment

Details

We perform a full matching on the NHANES_blood_lead_small dataset using the optmatch package. The code for constructing this matched dataset from the original dataset is given in the examples section. We add a column U0 as placeholder for the unmeasured confounder U.

Source

NHANES III, the Third US National Health and Nutrition Examination Survey.

References


Examples

```r
## Not run:
# To run this example, optmatch must be installed
set.seed(1)
library(optmatch)
data(NHANES_blood_lead_small)
attach(NHANES_blood_lead_small)

# Perform a fullmatch
fm = fullmatch(COP ~ ., data = NHANES_blood_lead_small[, 1:9], min.controls = 1/4, max.controls = 4)
NHANES_blood_lead_small_matched = cbind(NHANES_blood_lead_small, matches = fm)

# Add a U0 row
U0 = rep(1, dim(NHANES_blood_lead_small_matched)[1])
NHANES_blood_lead_small_matched = cbind(NHANES_blood_lead_small_matched[, 1:9], U0,
NHANES_blood_lead_small_matched[, 10:11])

## End(Not run)
```
Index

* datasets
  NHANES_blood_lead, 8
  NHANES_blood_lead_small, 10
  NHANES_blood_lead_small_matched, 11

calibrate_anim, 2
calibrate_one, 3
CI_block_boot, 4

EM_Algorithm, 5

find_border, 6
find_delta, 7

NHANES_blood_lead, 8
NHANES_blood_lead_small, 10
NHANES_blood_lead_small_matched, 11