Package 'surveybootstrap'

August 29, 2016

Title Tools for the Bootstrap with Survey Data

Version 0.0.1

Description Tools for using different kinds of bootstrap for estimating sampling variation using complex survey data.

License MIT + file LICENSE

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp

Depends plyr, dplyr, functional, stringr

Suggests knitr, testthat

RoxygenNote 5.0.1

NeedsCompilation yes

Author Dennis M. Feehan [aut, cre], Matthew J. Salganik [ths]

Maintainer Dennis M. Feehan <feehan@berkeley.edu>

Repository CRAN

Date/Publication 2016-05-04 12:14:27

R topics documented:

`bootstrap.estimates` *bootstrap.estimates*

Description

this function contains the core of the rescaled bootstrap method for estimating uncertainty in our estimates it should be designed so that it can be passed in to estimation functions as an argument OR

Usage

```
bootstrap.estimates(survey.data, survey.design, bootstrap.fn, estimator.fn,
  num.reps, weights = NULL, ..., summary.fn = NULL, verbose = TRUE,
  parallel = FALSE, paropts = NULL)
```

Arguments

`survey.data`	the dataset to use
`survey.design`	a formula describing the design of the survey (see below - TODO)
`bootstrap.fn`	name of the method to be used to take bootstrap resamples; see below
`estimator.fn`	name of a function which, given a dataset like survey data and arguments in `...`, will produce an estimate of interest
`num.reps`	the number of bootstrap replication samples to draw
`weights`	weights to use in estimation (or NULL, if none)
`...`	additional arguments which will be passed on to the estimator fn
`summary.fn`	(optional) name of a function which, given the set of estimates produced by estimator.fn, summarizes them. if not specified, all of the estimates are returned in a list
`verbose`	if TRUE, produce lots of feedback about what is going on
`parallel`	if TRUE, use the plyr library's .parallel argument to produce bootstrap resamples and estimates in parallel
`paropts`	if not NULL, additional arguments to pass along to the parallelization routine

Value

if no summary.fn is specified, then return the list of estimates produced by estimator.fn; if summary.fn is specified, then return its output

TODO

- estimator.fn/bootstrap.fn and summary.fn are treated differently (one expects characters, one expects an actual fn. fix!)
- write description block, including estimator.fn, bootstrap.fn, summary.fn, more?

Examples

```
# code goes here
```

---

chain.data *get a dataset from a chain*

Description

take the data for each member of the given chain and assemble it together in a dataset

Usage

```
chain.data(chain)
```

Arguments

chain	the chain to build a dataset from

Value

a dataset comprised of all of the chain's members' data put together. the order of the rows in the datset is not specified.

chain.size *get the size of a chain*

Description

count the total number of respondents in the chain and return it

Usage

```
chain.size(chain)
```

Arguments

chain	the chain object

Value

the number of respondents involved in the chain

chain.vals *chain.vals*

Description

get all of the values of the given variable found among members of a chain

Usage

```
chain.vals(chain, qoi.var = "uid")
```

Arguments

chain	the chain to get values from
qoi.var	the name of the variable to get from each member of the chain

Value

a vector with all of the values of `qoi.var` found in this chain. (currently, the order of the values in the vector is not guaranteed)

`estimate.degree.distns` *estimate degree distributions by trait*

Description

break down RDS degree distributions by trait, and return an object which has the degrees for each trait as well as functions to draw degrees from each trait.

Usage

```
estimate.degree.distns(survey.data, d.hat.vals, traits, keep.vars = NULL)
```

Arguments

`survey.data`	the respondent info
`d.hat.vals`	the variable that contains the degrees for each respondent
`traits`	a vector of the names of the columns of `survey.data` which refer to the traits
`keep.vars`	additional vars to return along with degrees

Details

one of the items returned as a result is a function, `draw.degrees.fn`, which takes one argument, `traits`. this is a vector of traits and, for each entry in this vector, `draw.degress.fn` returns a draw from the empirical distribution of degrees among respondents with that trait. so, `draw.degrees.fn(c("0.0", "0.1", "0.1")` would return a degree drawn uniformly at random from among the observed degrees of respondents with trait "0.0" and then two degrees from respondents with trait "0.1"

Value

an object with

- `distns` a list with one entry per trait value; each
- `draw.degrees.fn` a function which gets called with one
- `keep.vars` the name of the other vars that are kept (if any)

`estimate.mixing` *construct a mixing model from GoC/RDS data*

Description

given a dataset with the respondents and a dataset on the parents (in many cases the same individuals), and a set of relevant traits, estimate mixing parameters and return a markov model

Usage

```
estimate.mixing(survey.data, parent.data, traits)
```

Arguments

`survey.data`	the respondent info
`parent.data`	the parent info
`traits`	the names of the traits to build the model on

Value

a list with two entries:

- `mixing.df` the data used to estimate the mixing
- `choose.next.state.fn` a function which can be passed a vector of states and will return a draw of a subsequent state each entry in the vector
- `mixing.df` a dataframe (long-form) representation of the transition counts used to estimate the transition probabilities
- `states` a list with an entry for each state. within each state's entry are
  - `trans.probs` a vector of estimated transition probabilities
  - `trans.fn` a function which, when called, randomly chooses a next state with probabilities given by the transition probs.

`is.child.ct` *determine whether or not one id is a parent of another*

Description

this function allows us to determine which ids are directly descended from which other ones. it is the only part of the code that relies on the ID format used by the Curitiba study (TODO CITE); by modifying this function, it shold be possible to adapt this code to another study

Usage

```
is.child.ct(id, seed.id)
```

Arguments

id	the id of the potential child
seed.id	the id of the potential parent

Value

TRUE if id is the direct descendant of seed.id and FALSE otherwise

---

make.chain *build an RDS seed's chain from the dataset*

Description

text TODO assumes that the chain is a tree (no loops)

Usage

```
make.chain(seed.id, survey.data, is.child.fn = is.child.ct)
```

Arguments

seed.id	the id of the seed whose chain we wish to build from the dataset
survey.data	the dataset
is.child.fn	a function which takes two ids as arguments; it is expected to return TRUE if the second argument is the parent of the first, and FALSE otherwise. it defaults to is.child.ct

Value

info

---

max.depth *get the height (maximum depth) of a chain*

Description

get the height (maximum depth) of a chain

Usage

```
## S3 method for class 'depth'
max(chain)
```

Arguments

chain	the chain object

Value

the maximum depth of the chain

---

mc.sim *run a markov model*

Description

run a given markov model for n time steps, starting at a specified state

Usage

```
mc.sim(mm, start, n)
```

Arguments

mm	the markov model object returned by estimate.mixing
start	the name of the state to start in
n	the number of time-steps to run through

Details

this uses the markov model produced by estimate.mixing

Value

a vector with the state visited at each time step. the first entry has the starting state

---

MU284 *MU284 population*

Description

Data used in unit tests for variance estimation. See TODO-Sarndal TODO-sampling package TODO-doc describing unit tests

`rds.boot.draw.chain` *draw RDS bootstrap resamples for one chain*

Description

this function uses the algorithm described in the supporting online material for Weir et al 2012 (TODO PROPER CITE) to take bootstrap resamples of one chain from an RDS dataset

Usage

```
rds.boot.draw.chain(chain, mm, dd, parent.trait, idvar = "uid")
```

Arguments

`chain`	the chain to draw resamples for
`mm`	the mixing model to use
`dd`	the degree distns to use
`parent.trait`	a vector whose length is the number of bootstrap reps we want
`idvar`	the name of the variable used to label the columns of the output (presumably some id identifying the row in the original dataset they come from – see below)

Value

a list of dataframes with one entry for each respondent in the chain. each dataframe has one row for each bootstrap replicate. so if we take 10 bootstrap resamples of a chain of length 50, there will be 50 entries in the list that is returned. each entry will be a dataframe with 10 rows.

`rds.chain.boot.draws` *draw RDS bootstrap resamples*

Description

draw boostrap resamples for an RDS dataset, using the algorithm described in the supporting online material of Weir et al 2012 (TODO PROPER CITE)

Usage

```
rds.chain.boot.draws(chains, mm, dd, num.reps, keep.vars = NULL)
```

Arguments

chains	a list whose entries are the chains we want to resample
mm	the mixing model
dd	the degree distributions
num.reps	the number of bootstrap resamples we want
keep.vars	if not NULL, then the names of variables from the original dataset we want appended to each bootstrap resampled dataset (default is NULL)

Details

TODO – consider constructing chains, mm from other args

TODO be sure to comment the broken-out trait variables (ie these could all be different from the originals)

Value

a list of length num.reps; each entry in the list has one bootstrap-resampled dataset

---

rds.mc.boot.draws *draw RDS bootstrap resamples using the algorithm in Salganik 2006 (TODO PROPER CITE)*

Description

this algorithm picks a respondent from the survey to be a seed uniformly at random. it then generates a bootstrap draw by simulating the markov process forward for n steps, where n is the size of the draw required.

Usage

```
rds.mc.boot.draws(chains, mm, dd, num.reps)
```

Arguments

chains	a list with the chains constructed from the survey using make.chain
mm	the mixing model
dd	the degree distributions
num.reps	the number of bootstrap resamples we want

Details

if you wish the bootstrap dataset to end up with variables from the original dataset other than the traits and degree, then you must specify this when you construct dd using the 'estimate.degree.distns function.

TODO be sure to comment the broken-out trait variables (ie these could all be different from the originals)

Value

a list of length num.reps; each entry in the list has one bootstrap-resampled dataset

---

rescaled.bootstrap.sample
rescaled.bootstrap.sample

Description

C++ version: given a survey dataset and a description of the survey design (ie, which combination of vars determines primary sampling units, and which combination of vars determines strata), take a bunch of bootstrap samples for the rescaled bootstrap estimator (see, eg, Rust and Rao 1996).

Usage

```
rescaled.bootstrap.sample(survey.data, survey.design, parallel = FALSE,
  paropts = NULL, num.reps = 1)
```

Arguments

survey.data	the dataset to use
survey.design	a formula describing the design of the survey (see below - TODO)
parallel	if TRUE, use parallelization (via plyr)
paropts	an optional list of arguments passed on to plyr to control details of parallelization
num.reps	the number of bootstrap replication samples to draw

Details

Note that we assume that the formula uniquely specifies PSUs. This will always be true if the PSUs were selected without replacement. If they were selected with replacement, then it will be necessary to make each realization of a given PSU in the sample a unique id. Bottom line: the code below assumes that all observations within each PSU (as identified by the design formula) are from the same draw of the PSU.

The rescaled bootstrap technique works by adjusting the estimation weights based on the number of times each row is included in the resamples. If a row is never selected, it is still included in the returned results, but its weight will be set to 0. It is therefore important to use estimators that make use of the estimation weights on the resampled datasets.

We always take m_i = n_i - 1, according to the advice presented in Rao and Wu (1988) and Rust and Rao (1996).

survey.design is a formula of the form
weight ~ psu_vars + strata(strata_vars), where weight is the variable with the survey weights and psu is the variable denoting the primary sampling unit

Value

a list with num.reps entries. each entry is a dataset which has at least the variables index (the row index of the original dataset that was resampled) and weight.scale (the factor by which to multiply the sampling weights in the original dataset).

---

rescaled.bootstrap.sample.pureR

rescaled.bootstrap.sample.pureR

Description

(this is the pure R version; it has been supplanted by rescaled.bootstrap.sample, which is partially written in C++)

Usage

```
rescaled.bootstrap.sample.pureR(survey.data, survey.design, parallel = FALSE,
  paropts = NULL, num.reps = 1)
```

Arguments

survey.data	the dataset to use
survey.design	a formula describing the design of the survey (see below - TODO)
parallel	if TRUE, use parallelization (via plyr)
paropts	an optional list of arguments passed on to plyr to control details of parallelization
num.reps	the number of bootstrap replication samples to draw

Details

given a survey dataset and a description of the survey design (ie, which combination of vars determines primary sampling units, and which combination of vars determines strata), take a bunch of bootstrap samples for the rescaled bootstrap estimator (see, eg, Rust and Rao 1996).

Note that we assume that the formula uniquely specifies PSUs. This will always be true if the PSUs were selected without replacement. If they were selected with replacement, then it will be necessary to make each realization of a given PSU in the sample a unique id. Bottom line: the code below assumes that all observations within each PSU (as identified by the design formula) are from the same draw of the PSU.

The rescaled bootstrap technique works by adjusting the estimation weights based on the number of times each row is included in the resamples. If a row is never selected, it is still included in the returned results, but its weight will be set to 0. It is therefore important to use estimators that make use of the estimation weights on the resampled datasets.

We always take m_i = n_i - 1, according to the advice presented in Rao and Wu (1988) and Rust and Rao (1996).

survey.design is a formula of the form
weight ~ psu_vars + strata(strata_vars), where weight is the variable with the survey weights and psu is the variable denoting the primary sampling unit

Value

a list with num.reps entries. each entry is a dataset which has at least the variables index (the row index of the original dataset that was resampled) and weight.scale (the factor by which to multiply the sampling weights in the original dataset).

---

srs.bootstrap.sample *srs.bootstrap.sample*

Description

given a survey dataset and a description of the survey design (ie, which combination of vars determines primary sampling units, and which combination of vars determines strata), take a bunch of bootstrap samples under a simple random sampling (with repetition) scheme

Usage

```
srs.bootstrap.sample(survey.data, num.reps = 1, parallel = FALSE,
  paropts = NULL, ...)
```

Arguments

survey.data	the dataset to use
num.reps	the number of bootstrap replication samples to draw
parallel	if TRUE, use parallelization (via plyr)
paropts	an optional list of arguments passed on to plyr to control details of parallelization
...	ignored, but useful because it allows params like which are used in other bootstrap designs, to be passed in without error

Value

a list with num.reps entries. each entry is a dataset which has at least the variables index (the row index of the original dataset that was resampled) and weight.scale (the factor by which to multiply the sampling weights in the original dataset).

---

surveybootstrap *Survey bootstrap variance estimators*

Description

surveybootstrap has methods for analyzing data that were collected using network reporting techniques. It includes estimators appropriate for the simple boostrap and the rescaled bootstrap.

Index