Package 'networkreporting'

December 5, 2016

Title Tools for using Network Reporting Estimators

Description Functions useful
for producing estimates from data that were collected using network reporting techniques like network scale-up, indirect sampling, network reporting, and sibling history.

Version 0.1.1

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

Depends R (>= 2.11.0)

License MIT + file LICENSE

LazyData true

VignetteBuilder knitr

Imports functional, ggplot2, lazyeval, plyr, reshape2, stringr, dplyr, surveybootstrap

SystemRequirements GNU make

Suggests knitr, testthat

RoxygenNote 5.0.1

NeedsCompilation no

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

Repository CRAN

Date/Publication 2016-12-05 18:28:47

R topics documented:

add.kp 2
df.to.kpvec 3
estimate.error 4
example.knownpop.dat 5
example.survey 5
gwsm.estimator 7
kp.degree.estimator 7

kp.estimator_ 8
kp.individual.estimator 9
multiplicity.estimator 10
network.survival.estimator_ 11
networkreporting 13
nsum.estimator 14
nsum.internal.validation 15
plot_meanties_truth 16
rdsII.estimator 17
report.aggregator_ 18
summation.estimator 18
topcode.data 19
topcode.var 20
total.degree.estimator 21

Index **22**

`add.kp` *attach known populations to a dataframe*

Description

take a known population vector (see `df.to.kpvec`) and associate it with a survey dataframe. this makes it more convenient to use some of the `networksampling` package's function

Usage

```
add.kp(survey.data, kp.vec, total.pop.size = NULL)
```

Arguments

`survey.data`	the survey dataframe
`kp.vec`	the known population vector
`total.pop.size`	(optional) the total population size to use (see below)

Details

The `total.popn.size` parameter is interpreted as follows:

- NA if total.popn.size is NA then work with proportions
- NULL if total.popn.size is NULL (nothing passed in), then assume that there's a total.popn.size attribute associated with the dataset we're using
- numerical value if an actual total.popn.size was passed in, use that value

Value

the survey dataframe with the known population vector attached as an attribute

See Also

df.to.kpvec

Examples

```
## Not run:

  # if kp.dat is a dataframe with columns 'kp' with known popn names
  # and 'total.size' with the total size,
  # and my.survey is the dataframe with survey responses

  my.kp.vec <- df.to.kpvec(kp.data, kp.var='kp', kp.value='total.size')
  my.survey <- add.kp(my.survey, my.kp.vec)

  # now we can call estimator functions like
  # kp.degree.estimator without having to specify known
  # populations each time

## End(Not run)
```

---

df.to.kpvec *turn a dataframe into a known population vector*

Description

df.to.kpvec takes a dataframe which has a column with known population names, and a column with known population totals, and turns it into a known population vector. if the names of the survey variables corresponding to each known population are available, they can be passed in as well

Usage

```
df.to.kpvec(kp.data, kp.var, kp.value)
```

Arguments

kp.data	the known population dataset
kp.var	the column of kp.data that has known population names; either a column name, a column index, or a vector of values
kp.value	the column of kp.data that has known population sizes; either a column name, a column index, or a vector of value

Value

a vector whose entries have the known population values and whose names have the corresponding kp.var value

See Also

add.kp

Examples

```
## Not run:
  ## see example in add.kp

## End(Not run)
```

`estimate.error` *given an estimated subpopn size or prevalence and the correct value, produce some measurements of how close the esimate is*

Description

given an estimated subpopn size or prevalence and the correct value, produce some measurements of how close the esimate is

Usage

```
estimate.error(estimate, truth)
```

Arguments

`estimate`	the estimate
`truth`	the correct answer

Value

a vector whose entries have various summaries of fit

Examples

```
## Not run:
   ## TODO add example

## End(Not run)
```

example.knownpop.dat *Example known population data*

Description

Example of a household survey dataset, used in unit tests and vignettes for the networkreporting package.

Usage

```
example.knownpop.dat
```

Format

A data frame with 22 rows and 2 variables:

known.popn The name of the group

size The number of people in the group

example.survey *Example household survey data*

Description

Example of a household survey dataset, used in unit tests and vignettes for the networkreporting package.

Usage

```
example.survey
```

Format

A data frame with 2,406 rows and 36 variables:

id a unique identifier

cluster the cluster (part of the complex survey design)

region the region (part of the complex survey design)

indweight the individual weight (relative)

sex the sex of the respondent

age.cat the age category of the respondent

widower reported connections to widowers

nurse.or.doctor reported connections to nurses or doctors

male.community.health reported connections to male community health workers

teacher reported connections to teachers

woman.smoke reported connections to women who smoke

priest reported connections to priests

civil.servant reported connections to civil servants

woman.gave.birth reported connections to women who gave birth

muslim reported connections to Muslims

incarcerated reported connections to people who are incarcerated

judge reported connections to people who are judges

man.divorced reported connections to men who are divorced

treatedfortb reported connections to men who are divorced

nsengimana reported connections to Nsegimanas

murekatete reported connections to Murekatetes

twahirwa reported connections to Twahirwas

mukandekezi reported connections to Mukandekezis

nsabimana reported connections to Nsabimanas

mukamana reported connections to Mukamanas

ndayambaje reported connections to Ndayambajes

nyiraneza reported connections to Nyiranezas

bizimana reported connections to Bizimanas

nyirahabimana reported connections to Nyirahabimanas

ndagijimana reported connections to Ndagijimanas

mukandayisenga reported connections to Mukandayisengas

died reported connections to people who died

sex.workers reported connections to sex workers

msm reported connections to msm

idu reported connections to injecting drug users

clients reported connections to people who are clients of sex workers

gwsm.estimator *indirect estimator (generalized weight share method / gwsm)*

Description

compute gwsm estimate of the population size

Usage

```
gwsm.estimator(survey.data, gwsm.col = "mult")
```

Arguments

survey.data	the dataframe with the survey results
gwsm.col	the name or index of the column that contains, for each respondent, the individual value of the number known divided by the sum of the multiplicities (TODO MORE DETAIL)

Value

the multiplicity estimate of the hidden population's size (as a prevalence)

kp.degree.estimator *kp.degree.estimator (DEPRECATED)*

Description

see kp.individual.estimator instead.

Usage

```
kp.degree.estimator(survey.data, known.popns = NULL, total.popn.size = NULL,
  missing = "ignore", verbose = FALSE)
```

Arguments

survey.data	the dataframe with the survey results
known.popns	if not NULL, a vector whose entries are the size of the known populations, and whose names are the variable names in the dataset corresponding to each one. if NULL, then assume that the survey.data dataframe has an attribute called 'known.popns' containing this vector.
total.popn.size	the size of the entire population. if NULL, this function returns proportions; if not NULL, it returns absolute numbers (ie, the proportions * total popn size)

missing	if "ignore", then proceed with the analysis without doing anything about missing values. if "complete.obs" then, for each row, use only the known populations that have no missingness for the computations. care must be taken in using this second option
verbose	if TRUE, print messages to the screen

Details

compute an estimate of the respondents' degrees using the known population method

note that this function does not take survey weights, since these estimates are not for total degree, but just for the individual degree of each respondent

Value

a vector with an estimate of the degree for each row in survey.data. if missing=="ignore", then the degree for rows that have missingness in the 'how many X' questions will be set to NA

---

kp.estimator_ *Average personal network size estimates using known population method*

Description

If given `attribute.names`, then this function produces estimated average network sizes given by the groups that are defined by all combinations of the attributes; otherwise, it estimates the average personal network size for the entire frame population.

Usage

```
kp.estimator_(resp.data, known.populations, attribute.names, weights,
  total.kp.size = NULL, alter.popn.size = NULL)

kp.estimator(resp.data, known.populations, attribute.names, weights,
  total.kp.size = 1, alter.popn.size = NULL)
```

Arguments

resp.data	the dataframe that has the survey responses
known.populations	the names of the columns in `resp.data` that have respondents' reports about connections to known populations
attribute.names	the names of the columns in `resp.data` that determine the subgroups for which average degree is estimated
weights	weights to use in computing the estimate

`total.kp.size` the size of the probe alters; i.e., the sum of the known population sizes. if NULL, then this is set to 1

`alter.popn.size` the size of the population of alters; this is most often the frame population, which is the default if nothing else is specified; the size of the frame population is taken to be the sum of the weights over all of resp.data

Value

the estimated average degree for respondents in each of the categories given by `attribute.names`

Technical note

The estimated average degree is $(\sum y_{F_\alpha,A}/N_A) \times N_F/N_{F_\alpha}$ here, we estimate N_F/N_{F_α} by dividing the total of all respondents' weights by the sum of the weights for respondents in each cell α.

TODO

- handle case where attribute.names is NULL (should compute overall average)
- handle missing values
- integrate the individual-level estimator above, kp.degree.estimator
- finish documentation for NSE version
- make unit tests
- think about how to elegantly add options for dbar_(P,Q) vs dbar_(Q,P)

`kp.individual.estimator` *Individual personal network size estimates using the known population method*

Description

In most situations, the known population method will be used to estimate the average personal network size; this can be done with `kp.estimator_`. If, instead, you wish to estimate the personal network size of each individual respondent, then you can use this function.

Usage

```
kp.individual.estimator(resp.data, known.populations, total.kp.size = 1,
  alter.popn.size)

kp.individual.estimator_(resp.data, known.populations, total.kp.size = 1,
  alter.popn.size)
```

Arguments

`resp.data`	the respondent (survey) data
`known.populations`	the names of the known populations
`total.kp.size`	the sum of the sizes of all of the known populations
`alter.popn.size`	the size of the population respondents are reporting about connections to; typically this will be the frame population, so `alter.popn.size` should be the size of the frame population, N.F

Details

Note that this is not making inference about any larger population; it estimates a property of each individual respondent. So the sampling weights are not used here.

Value

a data frame with an estimate of each individual respondent's personal network size

TODO

- handle missing values!
- make unit tests

`multiplicity.estimator` *multiplicity.estimator*

Description

compute multiplicity estimate of the population size

Usage

```
multiplicity.estimator(survey.data, mult.col = "mult")
```

Arguments

`survey.data`	the dataframe with the survey results
`mult.col`	the name or index of the column that contains, for each respondent, the individual value of the number known divided by the sum of the multiplicities (TODO MORE DETAIL)

Value

the multiplicity estimate of the hidden population's size (as a prevalence)

network.survival.estimator_ *network survival estimator*

Description

use an aggregate multiplicity estimator and the respondents' own network size estimates to estimate hidden population sizes

Usage

```
network.survival.estimator_(resp.data, attribute.data, attribute.names,
  known.populations, total.kp.size = 1, weights, attribute.weights,
  dropmiss = NULL, verbose = TRUE)

network.survival.estimator(resp.data, attribute.data, attribute.names,
  known.populations, total.kp.size = 1, weights, attribute.weights,
  verbose = TRUE)
```

Arguments

resp.data	the dataframe that has a row for each respondent, with reported connections to the groups of known size, as well as the attributes. Note that the column names of the attributes should match their names in attribute.data
attribute.data	A dataframe with the reported attributes of hidden population members reported by survey respondents. There should be one row for each time a respondent reports a hidden population member. For example, to estimate death rates, there should be one row for each report of a death.
attribute.names	the names of the columns of attribute.data and resp.data that contain the attribute information.
known.populations	the names of the columns in resp.data that have responses to the known population questions
total.kp.size	the size of the probe alters, i.e., the sum of the known population sizes
weights	the weights or weights column for the respondent data
attribute.weights	the weights or weights column for the alter data
dropmiss	see report.aggregator
verbose	if TRUE, print information to screen

Details

This function takes two sources of data as input: first, it requires a long-form dataframe with the attributes of the reported members of the hidden population. For example, if we are asking about emigres and we collect the age and sex of each reported emigrant, then the long form dataset might look like:

age	sex	weight
15	m	2.10
58	f	1.15
33	m	3.67

The second source of data we need is the known population responses for the respondents, along with the *same* attributes for each respondent. For example, in the situation above, we would also require a dataset like this to be passed in

age	sex	weight	hm.teachers	hm.nurses	...
20	f	2.10	4	0	...
44	m	1.65	0	2	...
60	m	2.75	1	1	...

Value

the network reporting estimate of the hidden population's size (as a prevalence) broken down by the categories defined by all combinations of `attribute.names`.

Technical note

This function assumes that the sampling weights are standard analysis weights and *not* relative weights. Standard analysis weights should provide an estimate for the size of the frame population when added up; relative weights, on the other hand, will sum to the number of respondents in the sample. Demographic and Health surveys typically have relative weights, which must be converted into standard sampling weights before using this function.

TODO

- handle missing values
- think about whether or not this is the best way to handle N.F
- write more general agg mult est fn and call that
- make unit tests

`networkreporting` *Network reporting estimators*

Description

`networkreporting` has methods for analyzing data that were collected using network reporting techniques. It includes estimators appropriate for indirect sampling, network scale-up, network reporting, and sibling history methods.

nsum.estimator *nsum.estimator*

Description

compute network scale-up (nsum) estimate of the hidden population's size. if the degree ratio and information transmission rate are both 1 (the defaults), this is the Killworth estimator.

Usage

```
nsum.estimator(survey.data, d.hat.vals = "d", y.vals = "y",
  total.popn.size = NULL, deg.ratio = 1, tx.rate = 1, weights = NULL,
  killworth.se = FALSE, missing = "ignore", verbose = FALSE, ...)
```

Arguments

survey.data	the dataframe with survey results
d.hat.vals	the name or index of the column that contains each respondent's estimated degree
y.vals	the name or index of the column that contains the count of hidden popn members known
total.popn.size	NULL, NA, or a size
deg.ratio	the degree ratio, \frac\bard_T\bard; defaults to 1
tx.rate	the information transmission rate; defaults to 1
weights	if not NULL, weights to use in computing the estimate. this should be the name of the column in the survey.data which has the variable with the appropriate weights. these weights should be construted so that, eg, the mean of the degrees is estimated as (1/n) * \sum_i w_i * d_i
killworth.se	if not NA, return the Killworth et al estimate of
missing	if "ignore", then proceed with the analysis without doing anything about missing values. if "complete.obs" then only use rows that have no missingness for the computations (listwise deletion). care must be taken in using this second option
verbose	if TRUE, print messages to the screen
...	extra parameters to pass on to the bootstrap fn, if applicable

Details

TODO – cite Killworth estimator, our methods paper TODO – add refs to deg ratio and tx rate stuff...

Value

the nsum estimate of the hidden population's size (as a prevalence or an absolute number, depending on total.popn.size)

`nsum.internal.validation` *nsum.internal.validation*

Description

use a hold-one-out method to estimate the predictive accuracy of the network scale-up estimator on the known populations

Usage

```
nsum.internal.validation(survey.data, known.popns = NULL,
  total.popn.size = NULL, degrees = NULL, missing = "ignore",
  kp.method = FALSE, weights = NULL, killworth.se = FALSE,
  return.plot = FALSE, verbose = FALSE, bootstrap = FALSE, ...)
```

Arguments

`survey.data`	the dataframe with the survey results
`known.popns`	if not NULL, a vector whose entries are the size of the known populations, and whose names are the variable names in the dataset corresponding to each one. if NULL, then assume that the survey.data dataframe has an attribute called 'known.popns' containing this vector.
`total.popn.size`	the size of the entire population. if NA, this function works with proportions; if NULL, it looks for the 'total.popn.size' attribute of the dataset `survey.data`; if not NULL or NA, it works with absolute numbers (ie, the proportions * total popn size)
`degrees`	if not NULL, then the name or index of the column in the datset containing the degree estimates. if NULL, then use the known population method to estimate the degrees (see `kp.degree.estimator`)
`missing`	if "ignore", then proceed with the analysis without doing anything about missing values. if "complete.obs" then only use rows that have no missingness for the computations (listwise deletion). care must be taken in using this second option
`kp.method`	if TRUE, then we're using known population method estimates of the degrees. this means we have to recompute the degrees each time we hold out a known subgroup. if the degrees come from another estimator, like the summation method, then we don't need to do that since we don't use the ARD questions in coming up with the degree estimate.
`weights`	if not NULL, weights to use in computing the estimate. this should be the name of the column in the survey.data which has the variable with the appropriate weights. these weights should be construted so that, eg, the mean of the degrees is estimated as (1/n) * \sum_i w_i * d_i
`killworth.se`	if TRUE, return the Killworth et al estimate of the standard error
`return.plot`	if TRUE, make and return a ggplot2 plot object

verbose	if TRUE, report more detailed information about what's going on
bootstrap	if TRUE, use bootstrap.estimates to take bootstrap resamples in order to obtain intervals around each estimate. in this case, you are expected to also pass in at least bootstrap.fn, survey.design, and num.reps
...	additional arguments, which are passed on to bootstrap.estimates if bootstrap is TRUE

Details

given a set of estimated degrees, responses to a group of ARD questions, and the total size of the populations that the ARD questions ask about, this function estimates the accuracy of the network scale-up method by dropping each known population in turn, using the non-dropped populations to compute the degree and an estimate of the size of the known population, and comparing the result to the actual size of the known population

TODO – document bootstrap ci option better TODO – add example of usage to the comments... TODO – make amenable to parallelization

Value

a list with a dataset containing the subpopn-specific estimates, as well as several summaries of the accuracy of those estimates, including mae (mean absolute error), mse (mean squared error), rmse (root mean squared error), and are (average relative error)

---

plot_meanties_truth *plot_meanties_truth*

Description

plot the relationship between the mean number of ties in the survey dataset and the true popn sizes

Usage

```
plot_meanties_truth(survey.data, weights = NULL, known.popns = NULL)
```

Arguments

survey.data	the dataframe with the survey results
weights	if not NULL, weights to use in computing the estimate. this should be the name of the column in the survey.data which has the variable with the appropriate weights. these weights should be construted so that, eg, the mean of the degrees is estimated as (1/n) * \sum_i w_i * d_i
known.popns	if not NULL, a vector whose entries are the size of the known populations, and whose names are the variable names in the dataset corresponding to each one. if NULL, then assume that the survey.data dataframe has an attribute called 'known.popns' containing this vector.

Details

TODO - more in-depth description of this function

Value

a ggplot2 object with the relationship plot

`rdsII.estimator` *rdsII.estimator*

Description

compute an estimate for the prevalence of a trait from an RDS sample, using the estimator described in TODO [Volz + Heckathorn '08]

Usage

```
rdsII.estimator(survey.data, d.hat.vals, y.vals, missing = "ignore",
  verbose = FALSE)
```

Arguments

`survey.data`	the dataframe with RDS survey results
`d.hat.vals`	the name or index of the column that contains each respondent's estimated degree
`y.vals`	the name or index of the column that contains the quantity of interest. if this is a dichotomous trait, it should be 0 / 1
`missing`	if "ignore", then proceed with the analysis without doing anything about missing values. if "complete.obs" then only use rows that have no missingness for the computations (listwise deletion). care must be taken in using this second option
`verbose`	if TRUE, print messages to the screen

Details

NOTE: we have no weights for now, right? RDS doesn't get used with weights?

Value

the RDS-II estimate of the average of the quantity of interest

report.aggregator_ *aggregate a reported quantity by groups*

Description

This function takes a quantity and aggregates it by groups, using the design weights.

Usage

```
report.aggregator_(resp.data, attribute.names, qoi, weights, qoi.name,
  dropmiss = FALSE)

report.aggregator(resp.data, attribute.names = NULL, qoi, weights,
  qoi.name = NULL, dropmiss = FALSE)
```

Arguments

resp.data	the data
attribute.names	the names of the variables that define the groups for which the qoi should be aggregated
qoi	the variable with quantity to aggregate
weights	analysis weights
qoi.name	the name of the qoi
dropmiss	NOT YET IMPLEMENTED

Value

the estimated average degree for respondents in each of the categories given by attribute.names

summation.estimator *summation.estimator*

Description

compute an estimate of the respondents' degrees using the summation method

Usage

```
summation.estimator(survey.data, sum.q = NULL, missing = "ignore")
```

Arguments

survey.data	the dataframe with the survey results
sum.q	if not NULL, a vector whose entries are the variable names in the dataset corresponding to each summation question. if NULL, then assume that the survey.data dataframe has an attribute called 'sum.qs' containing this vector.
missing	if "ignore", then proceed with the analysis without doing anything about missing values. other options are not yet implemented.

Details

TODO – cite summation method ref

Note that the summation degree estimator for the case where there is missing data is not yet implemented. (In fact, I don't think that there is a known estimator for this case.)

Value

a vector with an estimate of the degree for each row in survey.data. if na.rm=TRUE, then the degree for rows that have missingness in the summation questions will be set to NA

---

topcode.data *topcode a group of variables*

Description

this function uses `topcode.var` to topcode a set of variables. it's useful for topcoding a whole set of aggregated relational data ("how many X are you connected to?") questions in the same way.

Usage

```
topcode.data(survey.data, vars, max, to.na = NULL, ignore = NA)
```

Arguments

survey.data	the dataset with the survey responses
vars	a vector with the names or indices of the columns in the dataframe that are to be topcoded
max	the maximum value; all values > max are recoded to max
to.na	a vector of values to recode to NA (this happens before topcoding)
ignore	a vector of values to leave unchanged

Value

the topcoded vector

Examples

```
## Not run:
   data(hh.survey) # example data included with the package
   example.survey <- topcode.data(example.survey,
                                  vars=known.popn.vars,
                                  max=30)

## End(Not run)
```

---

topcode.var *topcode a vector of numerical values*

Description

this function topcodes one vector; it's used by the topcode function to topcode a set of columns in a data frame

Usage

```
topcode.var(x, max, to.na = NULL, ignore = NA)
```

Arguments

x	the vector of values to topcode
max	the maximum value; all values > max are recoded to max
to.na	a vector of values to recode to NA (this happens before topcoding)
ignore	a vector of values to leave unchanged

Value

the topcoded vector

Examples

```
## Not run:
   ## TODO write example

## End(Not run)
```

`total.degree.estimator` *total.degree*

Description

estimate the total degree of the population network from sample degrees

Usage

```
total.degree.estimator(survey.data, d.hat.vals = "d", weights = NULL,
  missing = "ignore")
```

Arguments

`survey.data`	the dataframe with survey results
`d.hat.vals`	the name or index of the column that contains each respondent's estimated degree
`weights`	if not NULL, weights to use in computing the estimate. this should be the name of the column in the survey.data which has the variable with the appropriate weights. these weights should be construted so that, eg, the mean of the degrees is estimated as (1/n) * \sum_i w_i * d_i
`missing`	if "ignore", then proceed with the analysis without doing anything about missing values. if "complete.obs" then only use rows that have no missingness for the computations (listwise deletion). care must be taken in using this second option

Details

this computes the weighted sum of the respondents' estimated degrees.
' TODO – for now, it doesn't worry about missing values OR about differences between the frame and the universe

Value

the estimated total degree

Index

∗Topic **datasets**
example.knownpop.dat, 5
example.survey, 5

add.kp, 2, *4*

df.to.kpvec, *2, 3*, 3

estimate.error, 4
example.knownpop.dat, 5
example.survey, 5

gwsm.estimator, 7

kp.degree.estimator, 7, *15*
kp.estimator *(*kp.estimator_*)*, 8
kp.estimator_, 8, *9*
kp.individual.estimator, *7*, 9
kp.individual.estimator_
*(*kp.individual.estimator*)*, 9

multiplicity.estimator, 10

network.survival.estimator
*(*network.survival.estimator_*)*,
11
network.survival.estimator_, 11
networkreporting, 13
networkreporting-package
*(*networkreporting*)*, 13
nsum.estimator, 14
nsum.internal.validation, 15

package-networkreporting
*(*networkreporting*)*, 13
plot_meanties_truth, 16

rdsII.estimator, 17
report.aggregator, *11*
report.aggregator *(*report.aggregator_*)*,
18
report.aggregator_, 18

summation.estimator, 18

topcode.data, 19
topcode.var, 20
total.degree.estimator, 21