Package 'BMisc'

August 15, 2018

Title Miscellaneous Functions for Panel Data, Quantiles, and Printing Results

Date 2018-07-23

Version 1.3.1

Description These are miscellaneous functions for working with panel data, quantiles, and printing results. For panel data, the package includes functions for making a panel data balanced (that is, dropping missing individuals that have missing observations in any time period), converting id numbers to row numbers, and to treat repeated cross sections as panel data under the assumption of rank invariance. For quantiles, there are functions to make distribution functions from a set of data points (this is particularly useful when a distribution function is created in several steps), to combine distribution functions based on some external weights, and to invert distribution functions. Finally, there are several other miscellaneous functions for obtaining weighted means, weighted distribution functions, and weighted quantiles; to generate summary statistics and their differences for two groups; and to add or drop covariates from formulas.

Depends R (>= 2.1.0)

Imports formula.tools, plm

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Author Brantly Callaway [aut, cre]

Maintainer Brantly Callaway <brantly.callaway@temple.edu>

Repository CRAN

Date/Publication 2018-08-15 18:00:16 UTC

R topics documented:

`addCovToFormla` *addCovToFormla*

Description

`addCovFromFormla` adds some covariates to a formula; covs should be a list of variable names

Usage

```
addCovToFormla(covs, formla)
```

Arguments

`covs`	should be a list of variable names
`formla`	which formula to add covariates to

Value

formula

Examples

```
formla <- y ~ x
addCovToFormla(list("w","z"), formla)

formla <- ~x
addCovToFormla("z", formla)
```

BMisc *BMisc: A set of miscellaneous helper functions*

Description

BMisc: A set of miscellaneous helper functions

checkfun *checkfun*

Description

The check function used for optimizing to get quantiles

Usage

```
checkfun(a, tau)
```

Arguments

a	vector to compute quantiles for
tau	between 0 and 1, ex. .5 implies get the median

Value

numeric value

Examples

```
x <- rnorm(100)
x[which.min(checkfun(x, 0.5))] ##should be around 0
```

combineDfs *combineDfs*

Description

Combines two distribution functions with given weights by pstrat

Usage

```
combineDfs(y.seq, dflist, pstrat = NULL)
```

Arguments

y.seq	sequence of possible y values
dflist	list of distribution functions to combine
pstrat	a vector of weights to put on each distribution function; if weights are not provided then equal weight is given to each distribution function

Value

ecdf

Examples

```
x <- rnorm(100)
y <- rnorm(100,1,1)
Fx <- ecdf(x)
Fy <- ecdf(y)
both <- combineDfs(seq(-2,3,0.1), list(Fx,Fy))
plot(Fx, col="green")
plot(Fy, col="blue", add=TRUE)
plot(both, add=TRUE)
```

compareBinary *compareBinary*

Description

compareBinary ##takes in a variable e.g. union and runs bivariate regression of x on treatment (for summary statistics)

Usage

```
compareBinary(x, on, dta, w = rep(1, nrow(dta)), report = c("diff",
  "levels", "both"))
```

Arguments

x	variables to run regression on
on	binary variable
dta	the data to use
w	weights
report	which type of report to make; diff is the difference between the two variables by group

Value

matrix of results

---

cs2panel *cs2panel*

Description

Turn repeated cross sections data into panel data by imposing rank invariance; does not that the inputs have the same length

Usage

```
cs2panel(cs1, cs2, yname)
```

Arguments

cs1	data frame, the first cross section
cs2	data frame, the second cross section
yname	the name of the variable to calculate difference for (should be the same in each dataset)

Value

the change in outcomes over time

dropCovFromFormla *dropCovFromFormla*

Description

dropCovFromFormla adds drops some covariates from aformula; covs should be a list of variable names

Usage

```
dropCovFromFormla(covs, formla)
```

Arguments

covs	should be a list of variable names
formla	which formula to drop covariates from

Value

formula

Examples

```
formla <- y ~ x + w + z
dropCovFromFormla(list("w","z"), formla)

dropCovFromFormla("z", formla)
```

getWeightedDf *getWeightedDf*

Description

Get the mean applying some weights

Usage

```
getWeightedDf(y, y.seq = NULL, weights = NULL, norm = TRUE)
```

Arguments

y	a vector to compute the mean for
y.seq	an optional vector of values to compute the distribution function for; the default is to use all unique values of y
weights	the vector of weights, can be NULL, then will just return mean
norm	normalize the weights so that they have mean of 1, default is to normalize

Value

ecdf

`getWeightedMean` *getWeightedMean*

Description

Get the mean applying some weights

Usage

```
getWeightedMean(y, weights = NULL, norm = TRUE)
```

Arguments

`y`	a vector to compute the mean for
`weights`	the vector of weights, can be NULL, then will just return mean
`norm`	normalize the weights so that they have mean of 1, default is to normalize

Value

the weighted mean

`getWeightedQuantiles` *getWeightedQuantiles*

Description

Finds multiple quantiles by repeatedly calling getWeightedQuantile

Usage

```
getWeightedQuantiles(tau, cvec, weights = NULL, norm = TRUE)
```

Arguments

`tau`	a vector of values between 0 and 1
`cvec`	a vector to compute quantiles for
`weights`	the weights, weighted.checkfun normalizes the weights to sum to 1.
`norm`	normalize the weights so that they have mean of 1, default is to normalize

Value

vector of quantiles

`ids2rownum` *ids2rownum*

Description

ids2rownum takes a vector of ids and converts it t the right row number in the dataset; ids should be unique in the dataset that is, don't pass the function panel data with multiple same ids

Usage

```
ids2rownum(ids, data, idname)
```

Arguments

`ids`	vector of ids
`data`	data frame
`idname`	unique id

Value

vector of row numbers

Examples

```
ids <- seq(1,1000,length.out=100)
ids <- ids[order(runif(100))]
df <- data.frame(id=ids)
ids2rownum(df$id, df, "id")
```

`invertEcdf` *invertEcdf*

Description

take an ecdf object and invert it to get a step-quantile function

Usage

```
invertEcdf(df)
```

Arguments

`df`	an ecdf object

Value

stepfun object that contains the quantiles of the df

lhs.vars *lhs.vars*

Description

Take a formula and return a vector of the variables on the left hand side, it will return NULL for a one sided formula

Usage

```
lhs.vars(formla)
```

Arguments

formla	a formula

Value

vector of variable names

Examples

```
ff <- yvar ~ x1 + x2
lhs.vars(ff)
```

makeBalancedPanel *makeBalancedPanel*

Description

This function drops observations from data.frame that are not part of balanced panel data set.

Usage

```
makeBalancedPanel(data, idname, tname)
```

Arguments

data	data.frame used in function
idname	unique id
tname	time period name

Value

data.frame that is a balanced panel

Examples

```
id <- rep(seq(1,100,1),2) ## individual ids for setting up a two period panel
t <- rep(seq(1,2),100) ## time periods
y <- rnorm(200) ## outcomes
dta <- data.frame(id=id, t=t, y=y) ## make into data frame
dta <- dta[-7,] ## drop the 7th row from the dataset (which creates an unbalanced panel)
dta <- makeBalancedPanel(dta, idname="id", tname="t")
```

`makeDist` *makeDist*

Description

turn vectors of a values and their distribution function values into an ecdf. Vectors should be the same length and both increasing.

Usage

```
makeDist(x, Fx, sorted = FALSE, rearrange = FALSE)
```

Arguments

`x`	vector of values
`Fx`	vector of the distribution function values
`sorted`	boolean indicating whether or not x is already sorted; computation is somewhat faster if already sorted
`rearrange`	boolean indicating whether or not should monotize distribution function

Value

ecdf

Examples

```
y <- rnorm(100)
y <- y[order(y)]
u <- runif(100)
u <- u[order(u)]
F <- makeDist(y,u)
```

panel2cs *panel2cs*

Description

panel2cs takes a 2 period dataset and turns it into a cross sectional dataset. The default functionality is to keep all the variables from period 1 and add all the variables listed by name in timevars from period 2 to those

Usage

```
panel2cs(data, timevars, idname, tname)
```

Arguments

data	data.frame used in function
timevars	vector of names of variables to keep
idname	unique id
tname	time period name

Value

data.frame

rhs *rhs*

Description

Take a formula and return the right hand side of the formula

Usage

```
rhs(formla)
```

Arguments

formla	a formula

Value

a one sided formula

Examples

```
ff <- yvar ~ x1 + x2
rhs(ff)
```

rhs.vars *rhs.vars*

Description

Take a formula and return a vector of the variables on the right hand side

Usage

```
rhs.vars(formla)
```

Arguments

formla	a formula

Value

vector of variable names

Examples

```
ff <- yvar ~ x1 + x2
rhs.vars(ff)

ff <- y ~ x1 + I(x1^2)
rhs.vars(ff)
```

subsample *subsample*

Description

returns a subsample of a panel data set; in particular drops all observations that are not in keepids. If it is not set, randomly keeps nkeep observations.

Usage

```
subsample(dta, idname, tname, keepids = NULL, nkeep = NULL)
```

Arguments

dta	a data.frame which is a balanced panel
idname	the name of the id variable
tname	the name of the time variable
keepids	which ids to keep
nkeep	how many ids to keep (only used if keepids is not set); the default is the number of unique ids

Value

a data.frame that contains a subsample of `dta`

Examples

```
data(LaborSupply, package="plm")
nrow(LaborSupply)
unique(LaborSupply$year)
ss <- subsample(LaborSupply, "id", "year", nkeep=100)
nrow(ss)
```

`toformula` *toformula*

Description

take a name for a y variable and a vector of names for x variables and turn them into a formula

Usage

```
toformula(yname, xnames)
```

Arguments

`yname`	the name of the y variable
`xnames`	vector of names for x variables

Value

a formula

Examples

```
toformula("yvar", c("x1","x2"))

## should return yvar ~ 1
toformula("yvar", rhs.vars(~1))
```

`weighted.checkfun` *weighted.checkfun*

Description

Weights the check function

Usage

```
weighted.checkfun(q, cvec, tau, weights)
```

Arguments

`q`	the value to check
`cvec`	vector of data to compute quantiles for
`tau`	between 0 and 1, ex. .5 implies get the median
`weights`	the weights, weighted.checkfun normalizes the weights to sum to 1.

Value

numeric

Index