Package 'rata'

September 21, 2018

Title Manipulate Datasets Using A Stata-like Syntax

Version 0.0.1

Description This package implements tools for manipulating rectangular data sets (data sets with observations and variables) in a way that is familiar to users of a popular, but proprietary, statistical package commonly used in the social sciences.

Depends R (>= 3.5.1), Formula

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Author Zach Flynn [aut, cre]

Maintainer Zach Flynn <zlflynn@gmail.com>

R topics documented:

`collapse` *collapses a data set by variables using arbitrary aggregation functions*

Description

collapses a data set by variables using arbitrary aggregation functions

Usage

```
collapse(data, form)
```

Arguments

data	a data set. On exit, the dataset is the collapsed, aggregated dataset.
form	an argument with the form ~fun1(var1)+fun2(var2)+fun3(var3)+...|byvar1+byvar2+... . where fun1, fun2, and fun3 are aggregation functions like "mean", "sum", "max", etc. data will contain all unique levels of (byvar1,byvar2,...) and fun1(var1) conditional on each by level. In Stata, this is equivalent to collapse (fun1) var1 (fun2) var2 (fun3) var3 ..., by(byvar1 byvar2)

Examples

```
require(stats)
data <- cars
print(data)
collapse(data,~mean(dist)|speed)
print(data)
```

---

drop *drops variables or rows from the dataset*

Description

drops variables or rows from the dataset

Usage

```
drop(data, vars = NULL, rows = NULL)
```

Arguments

data	a data frame to drop variables from. On exit, the data frame will not have the variable or rows to drop.
vars	an optional argument with form, ~var1+var2+var3... Variables var1, var2, var3, ... are dropped. This works like the Stata command: "drop var1 var2 var3 ...".
rows	an optional argument which provides a condition that, if true, drops the row from the dataset. For example, drop(data,rows="female==1") would drop all women from the data set.

---

gen *generates a new variable that is a transformation of existing variables in the dataset or replaces one*

Description

generates a new variable that is a transformation of existing variables in the dataset or replaces one

Usage

```
gen(data, form, fun, subset = NULL, replace = FALSE)
```

Arguments

data	a data set. On exit, the dataset contains a newly generated variable or an existing variable is replaced.
form	this argument gives the name of the variable to generate, the variables that are used to create the new variable, and an option to generate the variable by a certain set of variables in the following format: newvar~var1+var2+var3|byvar1+byvar2.
fun	a function of the variables provided above, the value of which is put in newvar. For example, if newvar~var1+var2+var3|byvar1+byvar2 were specified and fun=function() sum(var1+var2+var3), then newvar would be the sum of var1, var2, and var3 for each level of (byvar1, byvar2). In Stata, the same command would be: "by byvar1 byvar2: gen newvar = sum(var1+var2+var3)".
subset	only generate values if the condition provided in subset is true. Make sure to enclose the expression in quotes, like so: subset="female==1 & highschool==1" to generate the values only for women who graduated from highschool. This option is used like the "if" in Stata.
replace	either TRUE or FALSE. If FALSE (default), the code refuses to alter the variable if the variable already exists. Otherwise, if replace=TRUE, then the values will be replaced.

---

keep *keeps some variables or rows in the dataset and drops the rest*

Description

keeps some variables or rows in the dataset and drops the rest

Usage

```
keep(data, vars = NULL, rows = NULL)
```

Arguments

data	a data frame. On exit, the data frame will not have the variable or rows that do not satisfy the keep conditions.
vars	an optional argument with form, ~var1+var2+var3... Variables var1, var2, var3, ... are kept. This works like the Stata command: "keep var1 var2 var3 ...".
rows	an optional argument which provides a condition that, if true, keeps the row in the dataset (otherwise, the row is dropped). For example, keep(data,rows="female==1") would keep all women in the data set and drop all men from it.

preserve *preserve a data set before modification*

Description

preserve a data set before modification

Usage

```
preserve(data)
```

Arguments

data	a data set to preserve

Value

a value that can be passed to `restore` to restore the data set later

Examples

```
require(stats)
data <- cars
p <- preserve(data)
collapse(data,~mean(dist)|speed)
data <- restore(p)
```

restore *restore a dataset from a previous preserve*

Description

restore a dataset from a previous preserve

Usage

```
restore(envir)
```

Arguments

envir	a previous preserve value.

Value

the preserved data set

Examples

```
require(stats)
data <- cars
p <- preserve(data)
collapse(data,~mean(dist)|speed)
data <- restore(p)
```

shape *reshapes a data set from wide to long or from long to wide formats*

Description

reshapes a data set from wide to long or from long to wide formats

Usage

```
shape(data, form, direction = "long")
```

Arguments

data	a data set. On exit, it will be reshaped.
form	if direction="long", then the argument should have the form: id1+id2+..~newvar|stub where there are variables in the data set named "stubXXXX" and "newvar" is the name of the new variable that will be added to the data set which will contain the various values of "stubXXXX" on exit. The variable "stub" on exit will contain the value of "XXXX". Variables (id1,id2,...) will also be included in the dataset on exit. The command behaves like "reshape long stub, i(id1 id2 ...) j(newvar)" in Stata. If direction="wide", then the argument should have the form, id1+id2+...~values1+values2+...|byvar1+byvar2+... The variables (id1,id2,...,byvar1,byvar2,...) should uniqely identify observations in the data. On exit the dataset will contain (id1,id2,...) in addition to values1byvar1.byvar2, values2byvar1.byvar2, ... for each unique value of (byvar1,byvar2,...). The command behaves like "reshape wide values1 values2 ..., i(id1 id2 ...) j(byvar1...)
direction	either "long" or "wide" to indicate the direction to reorient the data set

Index