help binsregtest

Title

binsregtest —— Data-driven Nonparametric Shape Restriction and Parametric Model Specification Testing using Binscatter.

Syntax

binsregtest *depvar* *indvar* [*covars*] [*if*] [*in*] [*weight*] [, **deriv(***v***)**
testmodel(*p s***)** **testmodelparfit(***filename***)** **testmodelpoly(***p***)**
testshape(*p s***)** **testshapel(***numlist***)** **testshaper(***numlist***)**
testshape2(*numlist***)**
bins(*p s***)** **nbins(***#***)** **binspos(***position***)** **binsmethod(***method***)** **nbinsrot(***#***)**
nsims(*#***)** **simsgrid(***#***)** **simsseed(***seed***)**
dfcheck(*n1 n2***)** **masspoints(***masspointsoption***)**
vce(*vcetype***)**]

where *depvar* is the dependent variable, *indvar* is the independent variable for binning, and *covars* are other covariates to be controlled for.

p, s and v are integers satisfying 0 <= s,v <= p, which can take different values in each case.

fweights, **aweight**s and **pweight**s are allowed; see weight.

Description

binsregtest implements binscatter-based hypothesis testing procedures for parametric functional forms and nonparametric shape restrictions on of the regression function estimators, following the results in Cattaneo, Crump, Farrell and Feng (2019a). If the binning scheme is not set by the user, the companion command binsregselect is used to implement binscatter in a data-driven (optimal) way and inference procedures are based on robust bias correction. Binned scatter plots can be constructed using the companion command binsreg.

A detailed introduction to this command is given in Cattaneo, Crump, Farrell and Feng (2019b). A companion R package with the same capabilities is available (see website below).

Companion commands: binsreg for binscatter estimation with robust inference procedures and plots, and binsregselect data-driven (optimal) binning selection.

Related Stata and R packages are available in the following website:

https://sites.google.com/site/nppackages/

Options

Estimand

deriv(*v***)** specifies the derivative order of the regression function for estimation, testing and plotting. The default is **deriv(0)**, which corresponds to the function itself.

Parametric Model Specification Testing

testmodel(*p s***)** sets a piecewise polynomial of degree *p* with *s* smoothness constraints for parametric model specification testing. The default is **testmodel(3 3)**, which corresponds to a cubic B-spline estimate of the regression function of interest for testing against the fitting from a parametric model specification.

testmodelparfit(*filename***)** specifies a dataset which contains the evaluation grid and fitted values of the model(s) to be tested against. The file must have a variable with the same name as *indvar*, which contains a series of evaluation points at which the binscatter model and the parametric model of interest are compared with each other. Each parametric model is represented by a variable named as *binsreg_fit**, which must contain the fitted values at the corresponding evaluation points.

testmodelpoly(*p***)** specifies the degree of a global polynomial model to be tested against.

Nonparametric Shape Restriction Testing

testshape(*p s***)** sets a piecewise polynomial of degree *p* with *s* smoothness constraints for nonparametric shape restriction testing. The default is **testmodel(3 3)**, which corresponds to a cubic B-spline estimate of the regression function of interest for one-sided or two-sided testing.

testshapel(*numlist***)** specifies a <u>numlist</u> of null boundary values for hypothesis testing. Each number *a* in the *numlist* corresponds to one boundary of a one-sided hypothesis test to the left of the form H0: *sup_x mu(x)<=a*.

testshaper(*numlist***)** specifies a <u>numlist</u> of null boundary values for hypothesis testing. Each number *a* in the *numlist* corresponds to one boundary of a one-sided hypothesis test to the right of the form H0: *inf_x mu(x)>=a*.

testshape2(*numlist***)** specifies a <u>numlist</u> of null boundary values for hypothesis testing. Each number *a* in the *numlist* corresponds to one boundary of a two-sided hypothesis test of the form H0: *sup_x |mu(x)-a|=0*.

Partitioning/Binning Selection

bins(*p s***)** sets a piecewise polynomial of degree *p* with *s* smoothness constraints for data-driven (IMSE-optimal) selection of the partitioning/binning scheme. The default is **bins(0 0)**, which corresponds to piecewise constant (canonical binscatter).

nbins(*#***)** sets the number of bins for partitioning/binning of *indvar*. If not specified, the number of bins is selected via the companion command <u>binsregselect</u> in a data-driven, optimal way whenever possible.

binspos(*position***)** specifies the position of binning knots. The default is **binspos(qs)**, which corresponds to quantile-spaced binning (canonical binscatter). Other options are: **es** for evenly-spaced binning, or a <u>numlist</u> for manual specification of the positions of inner knots (which must be within the range of *indvar*).

binsmethod(*method***)** specifies the method for data-driven selection of the number of bins via the companion command <u>binsregselect</u>. The default is **binsmethod(dpi)**, which corresponds to the IMSE-optimal direct plug-in rule. The other option is: **rot** for rule of thumb implementation.

nbinsrot(*#***)** specifies an initial number of bins value used to construct the DPI number of bins selector. If not specified, the data-driven ROT selector is used instead.

Simulation

nsims(*#***)** specifies the number of random draws for constructing confidence bands and hypothesis testing. The default is **nsims(500)**, which corresponds to 500 draws from a standard Gaussian random vector of size [(p+1)*J - (J-1)*s].

simsgrid(*#***)** specifies the number of evaluation points of an evenly-spaced grid within each bin used for evaluation of the supremum (or infimum) operation needed to construct confidence bands and hypothesis testing procedures. The default is **simsgrid(20)**, which corresponds to 20 evenly-spaced evaluation points within each bin for approximating the supremum (or infimum) operator.

simsseed(_#_**)** sets the seed for simulations.

Mass Points and Degrees of Freedom

dfcheck(_n1 n2_**)** sets cutoff values for minimum effective sample size checks, which take into account the number of unique values of _indvar_ (i.e., adjusting for the number of mass points), number of clusters, and degrees of freedom of the different statistical models considered. The default is **dfcheck(20 30)**. See Cattaneo, Crump, Farrell and Feng (2019b) for more details.

masspoints(_masspointsoption_**)** specifies how mass points in _indvar_ are handled. By default, all mass point and degrees of freedom checks are implemented. Available options:
masspoints(_noadjust_**)** omits mass point checks and the corresponding effective sample size adjustments.
masspoints(_nolocalcheck_**)** omits within-bin mass point and degrees of freedom checks.
masspoints(_off_**)** sets **masspoints(**_noadjust_**)** and **masspoints(**_nolocalcheck_**)** simultaneously.
masspoints(_veryfew_**)** forces the command to proceed as if _indvar_ has only a few number of mass points (i.e., distinct values). In other words, forces the command to proceed as if the mass point and degrees of freedom checks were failed.

Other Options

vce(_vcetype_**)** specifies the _vcetype_ for variance estimation used by the command regress. The default is **vce(robust)**.

Examples

Test linear model

```
    . binsregtest y x w, testmodelpoly(1)
```

Stored results

Scalars

e(N)	number of observations
e(Ndist)	number of distince values
e(Nclust)	number of clusters
e(nbins)	number of bins
e(p)	degree of polynomial for bin selection
e(s)	smoothness of polynomial for bin selection
e(testshape_p)	degree of polynomial for testing shape
e(testshape_s)	smoothnes of polynomial for testing shape
e(testmodel_p)	degree of polynomial for testing models
e(testmodel_s)	smoothness of polynomial for testing models
e(testpolyp)	degree of polynomial regression model
e(stat_poly)	statistic for testing global polynomial model
e(pval_poly)	p value for testing global polynomial model

Locals

e(testvalueL)	values in **testshapel()**
e(testvalueR)	values in **testshaper()**
e(testvalue2)	values in **testshape2()**
e(testvarlist)	varlist found in **testmodel()**

Matrices

e(stat_shapeL)	statistics for **testshapel()**
e(pval_shapeL)	p values for **testshapel()**
e(stat_shapeR)	statistics for **testshaper()**
e(pval_shapeR)	p values for **testshaper()**
e(stat_shape2)	statistics for **testshape2()**
e(pval_shape2)	p values for **testshape2()**
e(stat_model)	statistics for **testmodel()**
e(pval_model)	p values for **testmodel()**

References

Cattaneo, M. D., R. K. Crump, M. H. Farrell, and Y. Feng. 2019a. On Binscatter. *arXiv:1902.09608*.

Cattaneo, M. D., R. K. Crump, M. H. Farrell, and Y. Feng. 2019b. Binscatter Regressions. *arXiv:1902.09615*.

Authors

Matias D. Cattaneo, University of Michigan, Ann Arbor, MI. cattaneo@umich.edu.

Richard K. Crump, Federal Reserve Band of New York, New York, NY. richard.crump@ny.frb.org.

Max H. Farrell, University of Chicago, Chicago, IL. max.farrell@chicagobooth.edu.

Yingjie Feng, University of Michigan, Ann Arbor, MI. yjfeng@umich.edu.