

Appendix C: Stata Documentation for the psmatch2 command

This appendix contains the stata documentation for the psmatch2 routine. To obtain this collection of routines, type

```
ssc install psmatch2
```

from within stata.

Additional information on using the routine can be found on the website:

<http://fmwww.bc.edu/RePEc/usug2001/psmatch.pdf>

I have denoted by (***) the functions that are likely to be of most use for the Oportunidades evaluation. I recommend starting with nearest neighbor matching with a propensity score estimated by a logistic model and imposing the common support condition using both the common and "trim" options, with trimming set at a value in the range of 2-5%. I recommend using a biweight kernel function. I would not recommend using mahalanobis matching, because I have never seen any applications where this metric performed well. Also, I recommend matching with replacement. The matching can be performed at either the individual or household level.

```
help for psmatch2
```

```
-----
```

Mahalanobis and Propensity score Matching

```
(***) psmatch2 depvar [indepvars] [if exp] [in range] [, outcome(varlist)
pscore(varname)neighbor(integer) radius caliper(real) mahalanobis(varlist) add
pcaliper(real)kernel llr kerneltype(type) bwidth(real) common trim(real)
noreplacement descending odds index logit ties quietly w(matrix) ate]
```

Description

psmatch2 implements full Mahalanobis matching and a variety of propensity score matching methods to adjust for pre-treatment observable differences between a group of treated and a group of untreated. Treatment status is identified by depvar==1 for the treated and depvar==0 for the untreated observations.

psmatch2 is being continuously improved and developed. Make sure to keep your version up-to-date as follows

```
. ssc install psmatch2, replace
```

psmatch2 stores the estimate of the treatment effect on the treated in `r(ate)`, this allows easy bootstrapping of the standard error of the estimate. This can be done by simply executing:

```
. bs "psmatch2 training age gender, out(wage)" "r(ate)"
```

With `if` the average treatment is requested using option `ate` the estimate is returned in `r(ate)`. The average treatment effect on the untreated is then also returned in `r(atu)`. With more than one outcome variable the effects are returned as `r(ate_varname)` etc. for each outcome variable and effect.

See the documentation of `bs` for more details about bootstrapping in Stata.

If you want to be able to replicate your results you should set seed before calling `psmatch2`.

The propensity score - the conditional treatment probability - is either directly provided by the user or estimated by the program on the indepvars. Note that the sort order of your data could affect the results when using nearest-neighbor matching on a propensity score estimated with categorical (non-continuous) variables. Or more in general when there are untreated with identical propensity scores.

Matching methods to choose from are one-to-one (nearest neighbour or within caliper; with or without replacement), k-nearest neighbors, radius, kernel, local linear regression and Mahalanobis matching. The following list presents the syntax for each method.

If you have Stata 8 you can also [click here](#) to pop up a dialog or type `db psmatch2`.

Detailed Syntax

One-to-one matching:

```
psmatch2 depvar [indepvars] [if exp] [in range] [pweight] , [outcome(varlist)  
pscore(varname) mahalanobis(varlist) caliper(real) pcaliper(real) noreplacement  
descending common trim(real) odds index logit ties nowarnings quietly ate]
```

*** k-Nearest neighbors matching:

```
psmatch2 depvar [indepvars] [if exp] [in range] [pweight] , [outcome(varlist)  
pscore(varname) neighbor(integer k>1) caliper(real) common trim(real) odds  
index logit ties nowarnings quietly ate]
```

Radius matching:

```
psmatch2 depvar [indepvars] [if exp] [in range] [pweight] , radius caliper(real)  
[outcome(varlist) pscore(varname) common trim(real) odds index logit quietly ate]
```

*** Kernel matching:

```
psmatch2 depvar [indepvars] [if exp] [in range] [pweight] , kernel  
[outcome(varlist) psmatch2 depvar [indepvars] [if exp] [in range] [pweight] ,
```

```
kernel [outcome(varlist)kerneltype(kernel_type) pscore(varname) bwidth(real)
mahalanobis(varlist)common trim(real) odds index logit quietly atel]
```

*** Local linear regression matching:

```
psmatch2 depvar [indepvars] [if exp] [in range] [pweight] , llr outcome(varlist)
[kerneltype(kernel_type) pscore(varname) bwidth(real) mahalanobis(varlist)
common trim(real) odds index logit quietly atel]
```

Spline matching:

```
psmatch2 depvar [indepvars] [if exp] [in range] [pweight] , spline
outcome(varlist) [nknots(varname) pscore(varname) neighbor(integer) caliper(real)
common trim(real) odds index logit ties nowarnings quietly atel]
```

Mahalanobis matching:

```
psmatch2 depvar [if exp] [in range] [pweight] , mahalanobis(varlist) add
outcome(varlist) [kernel(kernel_type) llr bwidth(real) caliper(real) w(matrix) atel]
```

psmatch2 creates a number of variables for the convenience of the user:

`_treated` is a variable that equals 0 for control observations and 1 for treatment observations.

`_support` is an indicator variable with equals 1 if the observation is on the common support and 0 if the observation is off the support.

`_pscore` is the estimated propensity score or a copy of the one provided by `pscore()`.

`_outcome_variable` for every treatment observation stores the value of the matched outcome.

`_weight`. For nearest neighbor matching, it holds the frequency with which the observation is used as a match; with option `ties` and `k-nearest neighbors` matching it holds the normalized weight; for kernel matching, and `llr` matching with a weight other than `stata's tricube`, it stores the overall weight given to the matched observation. When estimating `att` only `_weight = 1` for the treated.

`_id` In the case of one-to-one and nearest-neighbors matching, a new identifier created for all observations.

`_n1` In the case of one-to-one and nearest-neighbors matching, for every treatment observation, it stores the observation number of the matched control observation if the data is sorted by `_id`. Do not forget to sort by `_id` if you want to use the observation number (`id`) of the nearest neighbor as in

```
. sort _id
. g x_of_match = x[_n1]
```

`_nn` In the case of nearest-neighbors matching, for every treatment observation, it stores the number of matched control observations.

`_pdif` In the case of one-to-one and nearest-neighbors matching, for every treatment observation, it stores the absolute distance to its matched control in terms of the propensity score.

`_mdif` In the case of one-to-one Mahalanobis matching, for every treatment observation it stores the absolute distance to its matched control in terms of the Mahalanobis metric.

Options

*** `outcome(varlist)` the outcome variable(s). When evaluating multiple outcomes `psmatch2` reduces to the min common number of observations with non-missing values on ALL outcomes, because otherwise the matching weights will not sum to the right number. If you have multiple outcomes with widely differing missing values you may wish to run `psmatch2` separately for each of the outcomes.

`ate` with this option the average treatment effect (`ate`) and average treatment effect on the untreated (`atu`) are reported in addition to the average treatment effect on the treated (`att`). The estimates are returned in `r(ate)`, `r(atu)` and `r(att)` respectively, see above.

Options: Estimation of the propensity score

`pscore(varname)` specifies the variable to be used as propensity score. Alternatively, `indepvars` need to be specified to allow the program to estimate the propensity score on them. In this case:

`logit` use logit instead of the default probit to estimate the propensity score.

`quietly` do not print output of propensity score estimation.

`odds` match on the odds ratio of the propensity score.

`index` use the latent variable index instead of the probability.

`nowarnings` do not test for control observations with duplicate propensity score values.

Options: Imposition of common support

*** `common` imposes a common support by dropping treatment observations whose `pscore` is higher than the maximum or less than the minimum `pscore` of the controls.

*** `trim(integer)` imposes common support by dropping # percent of the treatment observations at which the `pscore` density of the control observations is the lowest.

Options: Choice of matching estimator

*** neighbor(integer) number of neighbors used to calculate the matched outcome. Defaults to 1.

Default matching method is single nearest-neighbour (without caliper).
noreplacement perform 1-to-1 matching without replacement. Nearest neighbor propensity score matching only.

descending perform 1-to-1 matching without replacement in descending order.
Nearest neighbor propensity score matching only.

ties not only match nearest neighbor but also other controls with identical (tied) pcores.

radius perform radius matching within the specified radius given by caliper.

caliper(real) value for maximum distance of controls. Use to perform nearest neighbor(s) within caliper, radius matching and Mahalanobis 1-to-1 matching.

*** kernel perform kernel matching.

*** kerneltype(kernel_type) specifies the type of kernel:

normal the gaussian kernel.

*** biweight the biweight kernel.

epan the epanechnikov kernel (Default with kernel matching).

uniform the uniform kernel.

tricube the tricube kernel (Default with llr matching).

*** llr use local linear regression matching instead of kernel matching. Option kernel must be used to set the choice of kernel.

*** bwidth(real) the bandwidth for kernel and local linear regression matching. Default bandwidth is 0.06, except when doing local linear regression when the default bandwidth is 0.8. In this latter case centered subsets of $N \times \text{bwidth}$ observations are used.

mahalanobis(varlist) perform Mahalanobis-metric matching on varlist.

add adds the propensity score to varlist in mahalanobis(varlist).

pcaliper(real) See also caliper.

w(matrix) specify alternative weighting matrix. Mahalanobis-metric matching becomes matching on a quadratic metric with the specified weighting matrix.

spline performs 'spline-smoothing matching' by first fitting a natural cubic spline on pcore(or on the result from estimate) to outcome. The matched values are stored in the new variable, `_s_outcomevar`. (It requires the spline programme, which for stata7 needs to be downloaded by typing: `net install snp7_1.`)

nknots(integer) specifies the number of interior knots for spline smoothing. Default is the fourth root of the number of comparison units.

Examples

```
. psmatch2 training age gender, kernel k(biweight) out(wage)
. psmatch2 training age gender, n(5) logit
. psmatch2 training age gender, out(wage)
. bs "psmatch2 training age gender, out(wage)" "r(att)"
```

Also see

The commands `pstest`, `psgraph`.

Citing

`psmatch2` can be cited as follows

E. Leuven and B. Sianesi. (2003). "PSMATCH2: Stata module to perform full Mahalanobis and propensity score matching, common support graphing, and covariate imbalance testing".
<http://ideas.repec.org/c/boc/bocode/s432001.html>. Version 1.2.3.

Disclaimer

THIS SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY FORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT WILL THE COPYRIGHT HOLDERS OR THEIR EMPLOYERS, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THIS SOFTWARE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR O TY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Background Reading

Cochran, W. and Rubin, D.B. (1973), "Controlling Bias in Observational Studies", *Sankhya* 35, 417-446.

Dehejia, R.H and Wahba, S. (1999), "Causal Effects in Non-Experimental Studies: Re-Evaluating the Evaluation of Training Programmes", *Journal of the American Statistical Association* 94, 1053-1062.

Heckman, J.J., Ichimura, H. and Todd, P.E. (1997), "Matching As An Econometric Evaluation Estimator: Evidence from Evaluating a Job Training Programme", *Review of Economic Studies* 64, 605-654.

Heckman, J.J., Ichimura, H. and Todd, P.E. (1998), "Matching as an Econometric Evaluation Estimator", *Review of Economic Studies* 65, 261-294.

Heckman, J.J., Ichimura, H., Smith, J.A. and Todd, P. (1998), "Characterising Selection Bias Using Experimental Data", *Econometrica* 66, 5.

Heckman, J.J., LaLonde, R.J., Smith, J.A. (1998), "The Economics and Econometrics of Active Labour

Market Programmes", in Ashenfelter, O. and Card, D. (eds.), *The Handbook of Labour Economics* Vol.

3A.

Imbens, G. (2000), "The Role of Propensity Score in Estimating Dose-Response Functions", *Biometrika*

87(3), 706-710.

Lechner, M. (2001), Identification and Estimation of Causal Effects of Multiple Treatments under the

Conditional Independence Assumption, in: Lechner, M., Pfeiffer, F. (eds), *Econometric Evaluation of*

Labour Market Policies, Heidelberg/Springer, p. 43-58.

Rosenbaum, P.R. and Rubin, D.B. (1983), "The Central Role of the Propensity Score in Observational

Studies for Causal Effects", *Biometrika* 70, 1, 41-55.

Rosenbaum, P.R. and Rubin, D.B. (1985), "Constructing a Control Group Using Multivariate Matched

Sampling Methods that Incorporate the Propensity Score", *The American Statistician* 39(1), 33-38.

Rubin, D.B. (1974), "Estimating Causal Effects of Treatments in Randomised and Non-Randomised

Studies", *Journal of Educational Psychology* 66, 688-701.

Rubin, D.B. (1980), "Bias Reduction Using Mahalanobis-Metric Matching", *Biometrics* 36, 293-298.

Author

Edwin Leuven, Department of Economics, University of Amsterdam. If you observe any problems

<mailto:e.leuven@uva.nl>.

Barbara Sianesi, Institute for Fiscal Studies, London, UK.