Title

heckprobit postestimation — Postestimation tools for heckprobit

Postestimation commands   predict   margins   Remarks and examples
Also see

Postestimation commands

The following postestimation commands are available after `heckprobit`:

Command	Description
`contrast`	contrasts and ANOVA-style joint tests of estimates
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estat` (svy)	postestimation statistics for survey data
`estimates`	cataloging estimation results
`etable`	table of estimation results
* `hausman`	Hausman's specification test
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of coefficients
* `lrtest`	likelihood-ratio test
`margins`	marginal means, predictive margins, marginal effects, and average marginal effects
`marginsplot`	graph the results from margins (profile plots, interaction plots, etc.)
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
`predict`	probabilities, linear predictions and their SEs, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of estimates
`suest`	seemingly unrelated estimation
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.

predict

Description for predict

predict creates a new variable containing predictions such as probabilities, linear predictions, and standard errors.

Menu for predict

Statistics > Postestimation

Syntax for predict

predict [*type*] *newvar* [*if*] [*in*] [, *statistic* nooffset]

predict [*type*] *stub** [*if*] [*in*], scores

statistic	Description
Main	
pmargin	$\Phi(\mathbf{x}_j\mathbf{b})$, success probability; the default
p11	$\Phi_2(\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, \rho)$, predicted probability $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 1)$
p10	$\Phi_2(\mathbf{x}_j\mathbf{b}, -\mathbf{z}_j\mathbf{g}, -\rho)$, predicted probability $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 0)$
p01	$\Phi_2(-\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, -\rho)$, predicted probability $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 1)$
p00	$\Phi_2(-\mathbf{x}_j\mathbf{b}, -\mathbf{z}_j\mathbf{g}, \rho)$, predicted probability $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 0)$
psel	$\Phi(\mathbf{z}_j\mathbf{g})$, selection probability
pcond	$\Phi_2(\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, \rho)/\Phi(\mathbf{z}_j\mathbf{g})$, probability of success conditional on selection
xb	linear prediction
stdp	standard error of the linear prediction
xbsel	linear prediction for selection equation
stdpsel	standard error of the linear prediction for selection equation

$\Phi(\cdot)$ is the standard normal distribution function, and $\Phi_2(\cdot)$ is the bivariate normal distribution function.

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.

Options for predict

Main

pmargin, the default, calculates the univariate (marginal) predicted probability of success $\Pr(y_j^{\text{probit}} = 1)$.

p11 calculates the bivariate predicted probability $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 1)$.

p10 calculates the bivariate predicted probability $\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 0)$.

p01 calculates the bivariate predicted probability $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 1)$.

`p00` calculates the bivariate predicted probability $\Pr(y_j^{\text{probit}} = 0, y_j^{\text{select}} = 0)$.

`psel` calculates the univariate (marginal) predicted probability of selection $\Pr(y_j^{\text{select}} = 1)$.

`pcond` calculates the conditional (on selection) predicted probability of success
$\Pr(y_j^{\text{probit}} = 1, y_j^{\text{select}} = 1) / \Pr(y_j^{\text{select}} = 1)$.

`xb` calculates the probit linear prediction $\mathbf{x}_j\mathbf{b}$.

`stdp` calculates the standard error of the prediction, which can be thought of as the standard error of the predicted expected value or mean for the observation's covariate pattern. The standard error of the prediction is also referred to as the standard error of the fitted value.

`xbsel` calculates the linear prediction for the selection equation.

`stdpsel` calculates the standard error of the linear prediction for the selection equation.

`scores` calculates equation-level score variables.

The first new variable will contain $\partial \ln L / \partial(\mathbf{x}_j\boldsymbol{\beta})$.

The second new variable will contain $\partial \ln L / \partial(\mathbf{z}_j\boldsymbol{\gamma})$.

The third new variable will contain $\partial \ln L / \partial(\text{atanh}\,\rho)$.

`nooffset` is relevant only if you specified `offset(`*varname*`)` for `heckprobit`. It modifies the calculations made by `predict` so that they ignore the offset variable; the linear prediction is treated as $\mathbf{x}_j\mathbf{b}$ rather than as $\mathbf{x}_j\mathbf{b} + \text{offset}_j$.

margins

Description for margins

`margins` estimates margins of response for probabilities and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

`margins` [*marginlist*] [, *options*]

`margins` [*marginlist*] , `predict(`*statistic* ...`)` [`predict(`*statistic* ...`)` ...] [*options*]

statistic	Description
`pmargin`	$\Phi(\mathbf{x}_j\mathbf{b})$, success probability; the default
`p11`	$\Phi_2(\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, \rho)$, predicted probability $\Pr(y_j^{\mathrm{probit}} = 1, y_j^{\mathrm{select}} = 1)$
`p10`	$\Phi_2(\mathbf{x}_j\mathbf{b}, -\mathbf{z}_j\mathbf{g}, -\rho)$, predicted probability $\Pr(y_j^{\mathrm{probit}} = 1, y_j^{\mathrm{select}} = 0)$
`p01`	$\Phi_2(-\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, -\rho)$, predicted probability $\Pr(y_j^{\mathrm{probit}} = 0, y_j^{\mathrm{select}} = 1)$
`p00`	$\Phi_2(-\mathbf{x}_j\mathbf{b}, -\mathbf{z}_j\mathbf{g}, \rho)$, predicted probability $\Pr(y_j^{\mathrm{probit}} = 0, y_j^{\mathrm{select}} = 0)$
`psel`	$\Phi(\mathbf{z}_j\mathbf{g})$, selection probability
`pcond`	$\Phi_2(\mathbf{x}_j\mathbf{b}, \mathbf{z}_j\mathbf{g}, \rho)/\Phi(\mathbf{z}_j\mathbf{g})$, probability of success conditional on selection
`xb`	linear prediction
`xbsel`	linear prediction for selection equation
`stdp`	not allowed with `margins`
`stdpsel`	not allowed with `margins`

Statistics not allowed with `margins` are functions of stochastic quantities other than `e(b)`.

For the full syntax, see [R] **margins**.

Remarks and examples

stata.com

▷ Example 1

It is instructive to compare the marginal predicted probabilities with the predicted probabilities that we would obtain by ignoring the selection mechanism. To compare the two approaches, we will synthesize data so that we know the "true" predicted probabilities.

First, we need to generate correlated error terms, which we can do using a standard Cholesky decomposition approach. For our example, we will clear any data from memory and then generate errors that have a correlation of 0.5 by using the following commands. We set the seed so that interested readers can type in these same commands and obtain the same results.

```
. set seed 12309
. set obs 5000
Number of observations (_N) was 0, now 5,000.
. generate c1 = rnormal()
. generate c2 = rnormal()
. matrix P = (1,.5\.5,1)
. matrix A = cholesky(P)
. local fac1 = A[2,1]
. local fac2 = A[2,2]
. generate u1 = c1
. generate u2 = `fac1'*c1 + `fac2'*c2
```

We can check that the errors have the correct correlation by using the `correlate` command. We will also normalize the errors so that they have a standard deviation of one, so we can generate a bivariate probit model with known coefficients. We do that with the following commands:

```
. correlate u1 u2
(obs=5,000)

             |       u1       u2
-------------+------------------
          u1 |   1.0000
          u2 |   0.5012   1.0000

. summarize u1
  (output omitted)
. replace u1 = u1/r(sd)
(5,000 real changes made)
. summarize u2
  (output omitted)
. replace u2 = u2/r(sd)
(5,000 real changes made)
. drop c1 c2
. generate x1 = runiform()-.5
. generate x2 = runiform()+1/3
. generate y1s = 0.5 + 4*x1 + u1
. generate y2s = 3 - 3*x2 + .5*x1 + u2
. generate y1 = (y1s>0)
. generate y2 = (y2s>0)
```

We have now created two dependent variables, `y1` and `y2`, which are defined by our specified coefficients. We also included error terms for each equation, and the error terms are correlated. We run `heckprobit` to verify that the data have been correctly generated according to the model

$$y_1 = .5 + 4x_1 + u_1$$
$$y_2 = 3 + .5x_1 - 3x_2 + u_2$$

where we assume that y_1 is observed only if $y_2 = 1$.

```
. heckprobit y1 x1, sel(y2 = x1 x2) nolog
Probit model with sample selection              Number of obs     =      5,000
                                                      Selected    =      3,182
                                                      Nonselected =      1,818

                                                Wald chi2(1)      =     947.76
Log likelihood = -3612.401                      Prob > chi2       =     0.0000
```

	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
y1						
x1	4.015564	.130436	30.79	0.000	3.759914	4.271214
_cons	.4795158	.0471276	10.17	0.000	.3871473	.5718842
y2						
x1	.5361114	.0711951	7.53	0.000	.3965715	.6756513
x2	-3.017537	.0817541	-36.91	0.000	-3.177772	-2.857302
_cons	2.990145	.0765942	39.04	0.000	2.840024	3.140267
/athrho	.5339516	.0854577	6.25	0.000	.3664575	.7014457
rho	.4883959	.0650735			.3508892	.6052846

```
LR test of indep. eqns. (rho = 0): chi2(1) = 41.36        Prob > chi2 = 0.0000
```

Now that we have verified that we have generated data according to a known model, we can obtain and then compare predicted probabilities from the probit model with sample selection and a (usual) probit model.

```
. predict pmarg
(option pmargin assumed; Pr(y1=1))
. probit y1 x1 if y2==1
  (output omitted)
. predict phat
(option pr assumed; Pr(y1))
```

Using the (marginal) predicted probabilities from the probit model with sample selection (`pmarg`) and the predicted probabilities from the (usual) probit model (`phat`), we can also generate the "true" predicted probabilities from the synthesized `y1s` variable and then compare the predicted probabilities:

```
. generate ptrue = normal(y1s)
. summarize pmarg ptrue phat
```

Variable	Obs	Mean	Std. dev.	Min	Max
pmarg	5,000	.6089004	.3249993	.0632337	.99354
ptrue	5,000	.5967872	.3534232	2.78e-07	1
phat	5,000	.6588519	.3113716	.0910951	.997021

Here we see that ignoring the selection mechanism (comparing the `phat` variable with the true `ptrue` variable) results in predicted probabilities that are much higher than the true values. Looking at the marginal predicted probabilities from the model with sample selection, however, results in more accurate predictions.

◁

Also see

[R] **heckprobit** — Probit model with sample selection

[U] **20 Estimation and postestimation commands**