Title

eprobit predict — predict after eprobit

Description
Options for statistics
Remarks and examples
Also see
Syntax
Options for how results are calculated
Methods and formulas

Description

In this entry, we show how to create new variables containing observation-by-observation predictions after fitting a model with `eprobit`.

Syntax

You previously fit the model

```
eprobit y x1 ..., ...
```

The equation specified immediately after the `eprobit` command is called the main equation. It is

$$\Pr(y_i) = \Pr(\beta_0 + \beta_1 \texttt{x1}_i + \cdots + e_i.\texttt{y} > 0)$$

`predict` calculates predictions for $\Pr(\texttt{y})$ in the main equation. The other equations in the model are called auxiliary equations or complications.

The syntax of `predict` is

`predict` [*type*] *newvar* [*if*] [*in*] [`,` *stdstatistics howcalculated*]

stdstatistics	Description
`pr`	probability of positive outcome; the default
`xb`	linear prediction excluding all complications

howcalculated	Description
default	not fixed; base values from data
`fix(`*endogvars*`)`	fix specified endogenous covariates
`base(`*valspecs*`)`	specify base values of any variables
`target(`*valspecs*`)`	more convenient way to specify `fix()` and `base()`

Note: The `fix()` and `base()` options affect results only in models with endogenous variables in the main equation. The `target()` option is sometimes a more convenient way to specify the `fix()` and `base()` options.

endogvars are names of one or more endogenous variables appearing in the main equation.

valspecs specify the values for variables at which predictions are to be evaluated. Each *valspec* is of the form

varname = #

varname = (*exp*)

varname = *othervarname*

For instance, base(*valspecs*) could be base(w1=0) or base(w1=0 w2=1).

Notes:

(1) predict can also calculate treatment-effect statistics. See [ERM] **predict treatment**.

(2) predict can also make predictions for the other equations in addition to the main-equation predictions discussed here. See [ERM] **predict advanced**.

Options for statistics

pr calculates the predicted probability of a positive outcome. In each observation, the prediction is the probability conditioned on the covariates. Results depend on how complications are handled, which is determined by the *howcalculated* options.

xb specifies that the linear prediction be calculated ignoring all complications.

Options for how results are calculated

By default, predictions are calculated taking into account all complications. This is discussed in *Remarks and examples* of [ERM] **eregress predict**.

fix(*varname* ...) specifies a list of endogenous variables from the main equation to be treated as if they were exogenous. This was discussed in [ERM] **intro 3** and is discussed further in *Remarks and examples* of [ERM] **eregress predict**.

base(*varname* = ...) specifies a list of variables from any equation and values for them. If eprobit were a linear model, we would tell you those values will be used in calculating the expected value of e_i.y. That thinking will not mislead you but is not formally correct in the case of eprobit. Linear or nonlinear, errors from other equations spill over into the main equation because of correlations between errors. The correlations were estimated when the model was fit. The amount of spillover depends on those correlations and the values of the errors. This issue was discussed in [ERM] **intro 3** and is discussed further in *Remarks and examples* of [ERM] **eregress predict**.

target(*varname* = ...) is sometimes a more convenient way to specify the fix() and base() options. You specify a list of variables from the main equation and values for them. Those values override the values of the variables calculating $\beta_0 + \beta_1 \mathtt{x1}_i + \cdots$. Use of target() is discussed in *Remarks and examples* of [ERM] **eregress predict**.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Using predict after eprobit
How to think about nonlinear models

Using predict after eprobit

Predictions after fitting models with `eprobit` are handled the same as they are after fitting models with `eregress`. The issues are the same. See [ERM] **eregress predict**.

How to think about nonlinear models

Probit is a nonlinear model, and yet we just said that predictions after fitting models with `eprobit` are handled the same as they are after fitting models with `eregress`. That statement is partly true, not misleading, but false in its details.

The regression-base discussion that we routed you to is framed in terms of expected values. In the nonlinear models, it needs to be framed in terms of distributional assumptions about the errors. For instance, `predict` after `eprobit` does not predict the expected value (mean) of e_i`.y`. It calculates the probability that e_i`.y` exceeds $-\mathbf{x}_i\boldsymbol{\beta}$. These details matter hugely in implementation but can be glossed over for understanding the issues. For a full treatment of the issues, see *Methods and formulas* in [ERM] **eprobit**.

Methods and formulas

See *Methods and formulas* in [ERM] **eprobit postestimation**.

Also see

[ERM] **eprobit postestimation** — Postestimation tools for eprobit

[ERM] **eprobit** — Extended probit regression