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

**Syntax**

You previously fit the model

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

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

\[
\Pr(y_i) = \Pr(\beta_0 + \beta_1 x_{1i} + \cdots + e_i, y > 0)
\]

Or perhaps you had panel data and you fit the model with `xteprobit` by typing

```
xteprobit y x1 ... ...
```

Then the main equation would be

\[
\Pr(y_{ij}) = \Pr(\beta_0 + \beta_1 x_{1ij} + \cdots + u_i, y + v_{ij}, y > 0)
\]

In either case, `predict` calculates predictions for \(\Pr(y)\) in the main equation. The other equations in the model are called auxiliary equations or complications. Our discussion follows the cross-sectional case with a single error term, but it applies to the panel-data case when we collapse the random effects and observation-level error terms, \(e_{ij}.y = u_i. y + v_{ij}. y\).

The syntax of `predict` is

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

<table>
<thead>
<tr>
<th><code>stdstatistics</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pr</code></td>
<td>probability of positive outcome; the default</td>
</tr>
<tr>
<td><code>xb</code></td>
<td>linear prediction excluding all complications</td>
</tr>
</tbody>
</table>
**2 eprobit predict — predict after eprobit and xteprobit**

<table>
<thead>
<tr>
<th>howcalculated</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>not fixed; base values from data</td>
</tr>
<tr>
<td>fix(endogvars)</td>
<td>fix specified endogenous covariates</td>
</tr>
<tr>
<td>base(valspecs)</td>
<td>specify base values of any variables</td>
</tr>
<tr>
<td>target(valspecs)</td>
<td>more convenient way to specify fix() and base()</td>
</tr>
</tbody>
</table>

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 and xteprobit were fitting linear models, we would tell you those values will be used in calculating the expected value of $e_i$, $y$ (or $e_{ij}$, $y$ in the panel case). That thinking will not mislead you but is not formally correct in the case of eprobit and xteprobit. 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 $\beta_0 + \beta_1 x_1 + \cdots$. Use of target() is discussed in Remarks and examples of [ERM] eregress predict.

Remarks and examples

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 or xteprobit are handled the same as they are after fitting models with eregress and xteregress. 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 and xteprobit 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 $-x_i \beta$. 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 and xteprobit

[ERM] eprobit  —  Extended probit regression