

**nlogit postestimation** — Postestimation tools for nlogit

[Postestimation commands](#)

[predict](#)

[estat](#)

[Remarks and examples](#)

[Also see](#)

## Postestimation commands

The following postestimation command is of special interest after `nlogit`:

Command	Description
<a href="#">estat alternatives</a>	alternative summary statistics

The following standard postestimation commands are also available:

Command	Description
<a href="#">contrast</a>	contrasts and ANOVA-style joint tests of estimates
<a href="#">estat ic</a>	Akaike's and Schwarz's Bayesian information criteria (AIC and BIC)
<a href="#">estat summarize</a>	summary statistics for the estimation sample
<a href="#">estat vce</a>	variance–covariance matrix of the estimators (VCE)
<a href="#">estimates</a>	cataloging estimation results
<a href="#">hausman</a>	Hausman's specification test
<a href="#">lincom</a>	point estimates, standard errors, testing, and inference for linear combinations of coefficients
<a href="#">lrtest</a>	likelihood-ratio test
<a href="#">nlcom</a>	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
<a href="#">predict</a>	predictions, residuals, influence statistics, and other diagnostic measures
<a href="#">predictnl</a>	point estimates, standard errors, testing, and inference for generalized predictions
<a href="#">pwcompare</a>	pairwise comparisons of estimates
<a href="#">test</a>	Wald tests of simple and composite linear hypotheses
<a href="#">testnl</a>	Wald tests of nonlinear hypotheses

# predict

## Description for predict

`predict` creates a new variable containing predictions such as probabilities, linear predictions, conditional probabilities, and inclusive values.

## Menu for predict

Statistics > Postestimation

## Syntax for predict

```
predict [type] newvar [if] [in] [, statistic hlevel(#) altwise]
```

```
predict [type] {stub*|newvarlist} [if] [in], scores
```

*statistic*    Description

Main

<code>pr</code>	predicted probabilities of choosing the alternatives at all levels of the hierarchy or at level #, where # is specified by <code>hlevel(#)</code> ; the default
<code>xb</code>	linear predictors for all levels of the hierarchy or at level #, where # is specified by <code>hlevel(#)</code>
<code>condp</code>	predicted conditional probabilities at all levels of the hierarchy or at level #, where # is specified by <code>hlevel(#)</code>
<code>iv</code>	inclusive values for levels 2, ..., <code>e(levels)</code> or for <code>hlevel(#)</code>

The inclusive value for the first-level alternatives is not used in estimation; therefore, it is not calculated.

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

`pr` calculates the probability of choosing each alternative at each level of the hierarchy. Use the `hlevel(#)` option to compute the alternative probabilities at level #. When `hlevel(#)` is not specified,  $j$  new variables must be given, where  $j$  is the number of levels, or use `stub*` to have `predict` generate  $j$  variables with the prefix `stub` and numbered from 1 to  $j$ . The `pr` option is the default and if one new variable is given, the probability of the bottom-level alternatives are computed. Otherwise, probabilities for all levels are computed and `stub*` is still valid.

`xb` calculates the linear prediction for each alternative at each level. Use the `hlevel(#)` option to compute the linear predictor at level #. When `hlevel(#)` is not specified,  $j$  new variables must be given, where  $j$  is the number of levels, or use `stub*` to have `predict` generate  $j$  variables with the prefix `stub` and numbered from 1 to  $j$ .

`condp` calculates the conditional probabilities for each alternative at each level. Use the `hlevel(#)` option to compute the conditional probabilities of the alternatives at level #. When `hlevel(#)` is not specified,  $j$  new variables must be given, where  $j$  is the number of levels, or use `stub*` to have `predict` generate  $j$  variables with the prefix `stub` and numbered from 1 to  $j$ .

`iv` calculates the inclusive value for each alternative at each level. Use the `hlevel(#)` option to compute the inclusive value at level `#`. There is no inclusive value at level 1. If `hlevel(#)` is not used,  $j - 1$  new variables are required, where  $j$  is the number of levels, or use `stub*` to have `predict` generate  $j - 1$  variables with the prefix `stub` and numbered from 2 to  $j$ . See [Methods and formulas](#) in [R] `nlogit` for a definition of the inclusive values.

`hlevel(#)` calculates the prediction only for hierarchy level `#`.

`altwise` specifies that alternativewise deletion be used when marking out observations due to missing values in your variables. The default is to use casewise deletion. The `xb` option always uses alternativewise deletion.

`scores` calculates the scores for each coefficient in `e(b)`. This option requires a new-variable list of length equal to the number of columns in `e(b)`. Otherwise, use `stub*` to have `predict` generate enumerated variables with prefix `stub`.

## estat

### Description for estat

`estat alternatives` displays summary statistics about the alternatives in the estimation sample for each level of the tree structure.

### Menu for estat

Statistics > Postestimation

### Syntax for estat

```
estat alternatives
```

### Remarks and examples

[stata.com](http://www.stata.com)

`predict` may be used after `nlogit` to obtain the predicted values of the probabilities, the conditional probabilities, the linear predictions, and the inclusive values for each level of the nested logit model. Predicted probabilities for `nlogit` must be interpreted carefully. Probabilities are estimated for each case as a whole and not for individual observations.

#### ► Example 1

Continuing with our model in [example 3](#) of [R] `nlogit`, we refit the model and then examine a summary of the alternatives and their frequencies in the estimation sample.

```
. use http://www.stata-press.com/data/r14/restaurant
. nlogitgen type = restaurant(fast: Freebirds | MamasPizza,
> family: CafeEccell | LosNortenos | WingsNmore, fancy: Christophers | MadCows)
(output omitted)
. nlogit chosen cost rating distance || type: income kids, base(family) ||
> restaurant:, noconst case(family_id)
(output omitted)
```

```
. estat alternatives
```

```
Alternatives summary for type
```

index	Alternative value	label	Cases present	Frequency selected	Percent selected
1	1	fast	600	27	9.00
2	2	family	900	222	74.00
3	3	fancy	600	51	17.00

```
Alternatives summary for restaurant
```

index	Alternative value	label	Cases present	Frequency selected	Percent selected
1	1	Freebirds	300	12	4.00
2	2	MamasPizza	300	15	5.00
3	3	CafeEccell	300	78	26.00
4	4	LosNortenos	300	75	25.00
5	5	WingsNmore	300	69	23.00
6	6	Christophers	300	27	9.00
7	7	MadCows	300	24	8.00

Next we predict  $p_2 = \Pr(\text{restaurant})$ ;  $p_1 = \Pr(\text{type})$ ;  $\text{condp} = \Pr(\text{restaurant} \mid \text{type})$ ;  $\text{xb}_2$ , the linear prediction for the bottom-level alternatives;  $\text{xb}_1$ , the linear prediction for the first-level alternatives; and  $\text{iv}$ , the inclusive values for the bottom-level alternatives.

```
. predict p*
(option pr assumed)
. predict condp, condp hlevel(2)
. sort family_id type restaurant
. list restaurant type chosen p2 p1 condp in 1/14, sepby(family_id) divider
```

	restaurant	type	chosen	p2	p1	condp
1.	Freebirds	fast	1	.0642332	.1189609	.5399519
2.	MamasPizza	fast	0	.0547278	.1189609	.4600481
3.	CafeEccell	family	0	.284409	.7738761	.3675124
4.	LosNortenos	family	0	.3045242	.7738761	.3935051
5.	WingsNmore	family	0	.1849429	.7738761	.2389825
6.	Christophers	fancy	0	.0429508	.107163	.4007991
7.	MadCows	fancy	0	.0642122	.107163	.5992009
8.	Freebirds	fast	0	.0183578	.0488948	.3754559
9.	MamasPizza	fast	0	.030537	.0488948	.6245441
10.	CafeEccell	family	0	.2832149	.756065	.3745907
11.	LosNortenos	family	1	.3038883	.756065	.4019341
12.	WingsNmore	family	0	.1689618	.756065	.2234752
13.	Christophers	fancy	0	.1041277	.1950402	.533878
14.	MadCows	fancy	0	.0909125	.1950402	.466122

```
. predict xb*, xb
. predict iv, iv
```

```
. list restaurant type chosen xb* iv in 1/14, sepby(family_id) divider
```

	restaurant	type	chosen	xb1	xb2	iv
1.	Freebirds	fast	1	-1.124805	-1.476914	-.2459659
2.	MamasPizza	fast	0	-1.124805	-1.751229	-.2459659
3.	CafeEccell	family	0	0	-2.181112	.1303341
4.	LosNortenos	family	0	0	-2.00992	.1303341
5.	WingsNmore	family	0	0	-3.259229	.1303341
6.	Christophers	fancy	0	1.405185	-6.804211	-.745332
7.	MadCows	fancy	0	1.405185	-5.155514	-.745332
8.	Freebirds	fast	0	-1.804794	-2.552233	-.5104123
9.	MamasPizza	fast	0	-1.804794	-1.680583	-.5104123
10.	CafeEccell	family	0	0	-2.400434	.0237072
11.	LosNortenos	family	1	0	-2.223939	.0237072
12.	WingsNmore	family	0	0	-3.694409	.0237072
13.	Christophers	fancy	0	1.490775	-5.35932	-.6796131
14.	MadCows	fancy	0	1.490775	-5.915751	-.6796131

◀

## Also see

[R] [nlogit](#) — Nested logit regression

[U] [20 Estimation and postestimation commands](#)