Title

stata.com

discrim knn postestimation — Postestimation tools for discrim knn

Postestimation commands

The following postestimation commands are of special interest after `discrim knn`:

Command	Description
`estat classtable`	classification table
`estat errorrate`	classification error-rate estimation
`estat grsummarize`	group summaries
`estat list`	classification listing
`estat summarize`	estimation sample summary

The following standard postestimation commands are also available:

Command	Description
* `estimates`	cataloging estimation results
`predict`	group membership, probabilities of group membership, etc.

*All `estimates` subcommands except `table` and `stats` are available; see [R] **estimates**.

predict

Description for predict

`predict` creates a new variable containing predictions such as group classifications, probabilities, leave-one-out group classifications, and leave-one-out probabilities.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] *newvar* [*if*] [*in*] [, *statistic options*]

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

statistic	Description
Main	
`classification`	group membership classification; the default when one variable is specified and `group()` is not specified
`pr`	probability of group membership; the default when `group()` is specified or when multiple variables are specified
* `looclass`	leave-one-out group membership classification; may be used only when one new variable is specified
* `loopr`	leave-one-out probability of group membership

options	Description
Main	
`group(`*group*`)`	the group for which the statistic is to be calculated
Options	
`priors(`*priors*`)`	group prior probabilities; defaults to `e(grouppriors)`
`ties(`*ties*`)`	how ties in classification are to be handled; defaults to `e(ties)`
`noupdate`	do not update the within-group covariance matrix with leave-one-out predictions

priors	Description
`equal`	equal prior probabilities
`proportional`	group-size-proportional prior probabilities
matname	row or column vector containing the group prior probabilities
matrix_exp	matrix expression providing a row or column vector of the group prior probabilities

ties	Description
missing	ties in group classification produce missing values
random	ties in group classification are broken randomly
first	ties in group classification are set to the first tied group
nearest	ties in group classification are assigned based on the closest observation, or missing if this still results in a tie

You specify one new variable with `classification` or `looclass` and specify either one or `e(N_groups)` new variables with `pr` or `loopr`.

Unstarred statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample. Starred statistics are calculated only for the estimation sample, even when `if e(sample)` is not specified.

`group()` is not allowed with `classification` or `looclass`.

`noupdate` is an advanced option and does not appear in the dialog box.

Options for predict

Main

`classification`, the default, calculates the group classification. Only one new variable may be specified.

`pr` calculates group membership posterior probabilities. If you specify the `group()` option, specify one new variable. Otherwise, you must specify `e(N_groups)` new variables.

`looclass` calculates the leave-one-out group classifications. Only one new variable may be specified. Leave-one-out calculations are restricted to `e(sample)` observations.

`loopr` calculates the leave-one-out group membership posterior probabilities. If you specify the `group()` option, specify one new variable. Otherwise, you must specify `e(N_groups)` new variables. Leave-one-out calculations are restricted to `e(sample)` observations.

`group(`*group*`)` specifies the group for which the statistic is to be calculated and can be specified using

`#1`, `#2`, ..., where `#1` means the first category of the `e(groupvar)` variable, `#2` the second category, etc.;

the values of the `e(groupvar)` variable; or

the value labels of the `e(groupvar)` variable if they exist.

`group()` is not allowed with `classification` or `looclass`.

Options

`priors(`*priors*`)` specifies the prior probabilities for group membership. If `priors()` is not specified, `e(grouppriors)` is used. The following *priors* are allowed:

`priors(equal)` specifies equal prior probabilities.

`priors(proportional)` specifies group-size-proportional prior probabilities.

`priors(`*matname*`)` specifies a row or column vector containing the group prior probabilities.

`priors(`*matrix_exp*`)` specifies a matrix expression providing a row or column vector of the group prior probabilities.

`ties(`*ties*`)` specifies how ties in group classification will be handled. If `ties()` is not specified, `e(ties)` is used. The following *ties* are allowed:

`ties(missing)` specifies that ties in group classification produce missing values.

`ties(random)` specifies that ties in group classification are broken randomly.

`ties(first)` specifies that ties in group classification are set to the first tied group.

`ties(nearest)` specifies that ties in group classification are assigned based on the closest observation, or missing if this still results in a tie.

The following option is available with `predict` after `discrim knn` but is not shown in the dialog box:

`noupdate` causes the within-group covariance matrix not to be updated with leave-one-out predictions. `noupdate` is an advanced, rarely used option that is valid only if a Mahalanobis transformation is specified.

Remarks and examples

stata.com

kth-nearest-neighbor (KNN) discriminant analysis and postestimation can be time consuming for large datasets. The training data must be retained and then searched to find the nearest neighbors each time a classification or prediction is performed.

You can find more examples of postestimation with KNN in [MV] **discrim knn**, and more examples of the common `estat` subcommands in [MV] **discrim estat**.

▷ Example 1: Leave-one-out classification after KNN

Recall example 1 of [MV] **discrim knn**. We use a similar idea here, creating a two-dimensional dataset on the plane with x and y variables in $[-4, 4]$. Instead of random data, we choose data on a regular grid to make things easy to visualize, and once again, we assign groups on the basis of geometric calculations. To start, we assign all points a group value of one, then within four circles of radius 3, one in each quadrant, we change the group value to two in the circles in the first and third quadrants, and we change the group value to three in the circles in the second and fourth quadrants.

Instructions for creating this dataset and definitions for local macros associated with it are contained in its notes.

```
. use https://www.stata-press.com/data/r17/circlegrid
(Gridded circle data)
. local rp: di %12.10f 2+sqrt(3)
. local rm: di %12.10f 2-sqrt(3)
. local functionplot
> (function y =  sqrt(3-(x+2)^2) - 2, lpat(solid) range(-`rp' -`rm'))
> (function y = -sqrt(3-(x+2)^2) - 2, lpat(solid) range(-`rp' -`rm'))
> (function y =  sqrt(3-(x-2)^2) + 2, lpat(solid) range(-`rm'  `rp'))
> (function y = -sqrt(3-(x-2)^2) + 2, lpat(solid) range(-`rm'  `rp'))
> (function y =  sqrt(3-(x+2)^2) + 2, lpat(solid) range(-`rp' -`rm'))
> (function y = -sqrt(3-(x+2)^2) + 2, lpat(solid) range(-`rp' -`rm'))
> (function y =  sqrt(3-(x-2)^2) - 2, lpat(solid) range( `rm'  `rp'))
> (function y = -sqrt(3-(x-2)^2) - 2, lpat(solid) range( `rm'  `rp'))
. local graphopts
> aspectratio(1) legend(order(1 "group 1" 2 "group 2" 3 "group 3") rows(1))
```

```
. twoway (scatter y x if group==1)
>        (scatter y x if group==2)
>        (scatter y x if group==3)
>        'functionplot' , 'graphopts' name(original, replace)
>        title("Training data")
```

We do a KNN discriminant analysis, choosing `k(3)`. We elect to omit the standard classification table and instead take a look at the leave-one-out (LOO) classification table.

```
. discrim knn x y, group(group) k(3) priors(proportional) notable lootable
Kth-nearest-neighbor discriminant analysis
Leave-one-out classification summary
```

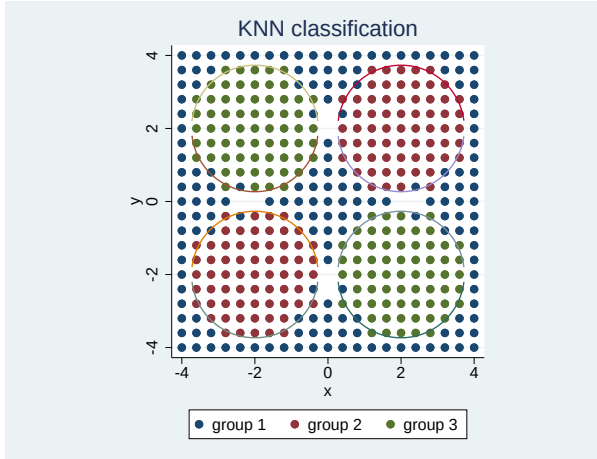
Key
Number Percent

True group	Classified 1	2	3	Total
1	173	12	12	197
	87.82	6.09	6.09	100.00
2	8	114	0	122
	6.56	93.44	0.00	100.00
3	8	0	114	122
	6.56	0.00	93.44	100.00
Total	189	126	126	441
	42.86	28.57	28.57	100.00
Priors	0.4467	0.2766	0.2766	

We will predict the LOO classification, changing to `priors(equal)`, and look at the plot.

```
. predict cknn, looclass priors(equal)
warning: 8 ties encountered;
         ties are assigned to missing values.
(8 missing values generated)
```

```
. twoway (scatter y x if cknn==1)
>        (scatter y x if cknn==2)
>        (scatter y x if cknn==3)
>        'functionplot' , 'graphopts' name(KNN, replace)
>        title("KNN classification")
```



We see several empty spots on the grid. In our output, changing to `priors(equal)` created several ties that were assigned to missing values. Missing values are the blank places in our graph.

◁

▷ Example 2: Listing misclassified observations

Continuing where we left off, we use `estat list` to display LOO probabilities for the misclassified observations, but this produces a lot of output.

```
. estat list, misclass class(noclass looclass) pr(nopr loopr) priors(equal)
```

	Classification		LOO Probabilities		
Obs	True	LOO Cl.	1	2	3
24	1	2 *	0.3836	0.6164	0.0000
28	1	2 *	0.2374	0.7626	0.0000
34	1	3 *	0.2374	0.0000	0.7626
38	1	3 *	0.3836	0.0000	0.6164
50	2	1 *	0.5513	0.4487	0.0000
54	3	1 *	0.5513	0.0000	0.4487

(*output omitted*)

```
* indicates misclassified observations
```

Instead, we predict the LOO probabilities and list only those where the LOO classification is missing.

```
. predict pr*, loopr priors(equal)
. list group cknn pr* if missing(cknn)
```

	group	cknn	pr1	pr2	pr3
94.	1	.	.2373541	.381323	.381323
115.	1	.	.2373541	.381323	.381323
214.	1	.	.2373541	.381323	.381323
215.	1	.	.2373541	.381323	.381323
225.	1	.	.2373541	.381323	.381323
226.	1	.	.2373541	.381323	.381323
325.	1	.	.2373541	.381323	.381323
346.	1	.	.2373541	.381323	.381323

The missing LOO classifications represent ties for the largest probability.

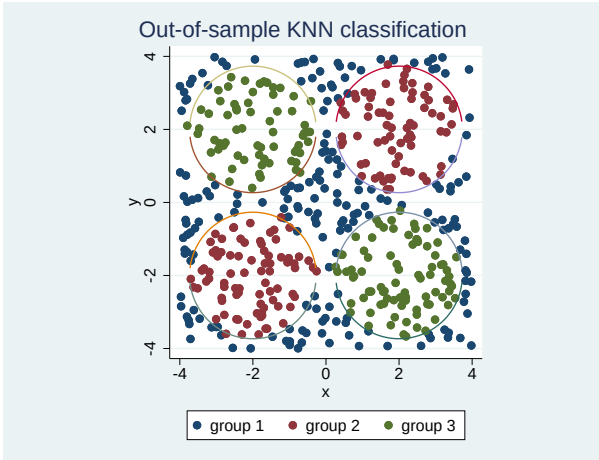
◁

▷ Example 3: Out-of-sample KNN classification

LOO classification and LOO probabilities are available only in sample, but standard probabilities can be obtained out of sample. To demonstrate this, we continue where we left off, with the KNN model of example 2 still active. We drop our current data and generate some new data. We predict the standard classification with the new data and graph our results.

```
. clear
. set obs 500
Number of observations (_N) was 0, now 500.
. set seed 314159265
. generate x = 8*runiform() - 4
. generate y = 8*runiform() - 4
. predict cknn, class
```

```
. twoway (scatter y x if cknn==1)
>        (scatter y x if cknn==2)
>        (scatter y x if cknn==3)
>        ‘functionplot’, ‘graphopts’ name(KNN2, replace)
>        title("Out-of-sample KNN classification", span)
```



Methods and formulas

See [MV] **discrim knn** for methods and formulas.

Also see

[MV] **discrim knn** — kth-nearest-neighbor discriminant analysis

[MV] **discrim estat** — Postestimation tools for discrim

[MV] **discrim** — Discriminant analysis

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