

forecast drop — Drop forecast variables

Description
Stored results

Quick start
Also see

Syntax

Options

Remarks and examples

Description

`forecast drop` drops variables previously created by `forecast solve`.

Quick start

Remove all variables created by `forecast solve` from the current dataset

```
forecast drop
```

Remove only forecast variables starting with `f_`

```
forecast drop, prefix(f_)
```

Remove only forecast variables ending with `_f`

```
forecast drop, suffix(_f)
```

Syntax

```
forecast drop [ , options ]
```

options

Description

* prefix(*string*) specify prefix for forecast variables

* suffix(*string*) specify suffix for forecast variables

* You can specify `prefix()` or `suffix()` but not both.

Options

`prefix(string)` and `suffix(string)` specify either a name prefix or a name suffix that will be used to identify forecast variables to be dropped. You may specify `prefix()` or `suffix()` but not both. By default, `forecast drop` removes all forecast variables produced by the previous invocation of `forecast solve`.

Suppose, however, that you previously specified the `simulate()` option with `forecast solve` and wish to remove variables containing simulation results but retain the variables containing the point forecasts. Then you can use the `prefix()` or `suffix()` option to identify the simulation variables you want dropped.

Remarks and examples

For an overview of the forecast commands, see [TS] [forecast](#). This manual entry assumes you have already read that manual entry. `forecast drop` safely removes variables previously created using `forecast solve`. Say you previously solved your model and created forecast variables that were suffixed with `_f`. Do not type

```
. drop *_f
```

to remove those variables from the dataset. Rather, type

```
. forecast drop
```

The former command is dangerous: Suppose you were given the dataset and asked to produce the forecast. The person who previously worked with the dataset created other variables that ended with `_f`. Using `drop` would remove those variables as well. `forecast drop` removes only those variables that were previously created by `forecast solve` based on the model in memory.

If you do not specify any options, `forecast drop` removes all the forecast variables created by the current model, including the variables that contain the point forecasts as well as any variables that contain simulation results specified by the `simulate()` option with `forecast solve`. Suppose you had typed

```
. forecast solve, prefix(s_) simulate(betas, statistic(stddev, prefix(sd_)))
```

Then if you type

```
. forecast drop, prefix(sd_)
```

`forecast drop` will remove the variables containing the standard deviations of the forecasts and will leave the variables containing the point forecasts (prefixed with `s_`) untouched.

`forecast drop` does not exit with an error if a variable it intends to drop does not exist in the dataset.

Stored results

`forecast drop` stores the following in `r()`:

Scalars

<code>r(n_dropped)</code>	number of variables dropped
---------------------------	-----------------------------

Also see

[TS] [forecast](#) — Econometric model forecasting

[TS] [forecast solve](#) — Obtain static and dynamic forecasts