Title

stata.com

mi impute logit — Impute using logistic regression

Description

`mi impute logit` fills in missing values of a binary variable by using a logistic regression imputation method. You can perform separate imputations on different subsets of the data by specifying the `by()` option. You can also account for frequency, importance, and sampling weights.

Menu

Statistics > Multiple imputation

Syntax

`mi impute logit` *ivar* [*indepvars*] [*if*] [*weight*] [, *impute_options* *options*]

impute_options	Description
Main	
* `add(#)`	specify number of imputations to add; required when no imputations exist
* `replace`	replace imputed values in existing imputations
`rseed(#)`	specify random-number seed
`double`	store imputed values in double precision; the default is to store them as `float`
`by(`*varlist*[, *byopts*]`)`	impute separately on each group formed by *varlist*
Reporting	
`dots`	display dots as imputations are performed
`noisily`	display intermediate output
`nolegend`	suppress all table legends
Advanced	
`force`	proceed with imputation, even when missing imputed values are encountered
`noupdate`	do not perform `mi update`; see [MI] **noupdate option**

*`add(#)` is required when no imputations exist; `add(#)` or `replace` is required if imputations exist.
`noupdate` does not appear in the dialog box.

options	Description
Main	
noconstant	suppress constant term
offset(*varname*)	include *varname* in model with coefficient constrained to 1
augment	perform augmented regression in the presence of perfect prediction
conditional(*if*)	perform conditional imputation
bootstrap	estimate model parameters using sampling with replacement
Maximization	
maximize_options	control the maximization process; seldom used

You must `mi set` your data before using `mi impute logit`; see [MI] **mi set**.

You must `mi register` *ivar* as imputed before using `mi impute logit`; see [MI] **mi set**.

indepvars may contain factor variables; see [U] **11.4.3 Factor variables**.

`collect` is allowed; see [U] **11.1.10 Prefix commands**.

`fweights`, `iweights`, and `pweights` are allowed; see [U] **11.1.6 weight**.

Options

Main

`noconstant`; see [R] **Estimation options**.

`add()`, `replace`, `rseed()`, `double`, `by()`; see [MI] **mi impute**.

`offset(`*varname*`)`; see [R] **Estimation options**.

`augment` specifies that augmented regression be performed if perfect prediction is detected. By default, an error is issued when perfect prediction is detected. The idea behind the augmented-regression approach is to add a few observations with small weights to the data during estimation to avoid perfect prediction. See *The issue of perfect prediction during imputation of categorical data* under *Remarks and examples* in [MI] **mi impute** for more information. `augment` is not allowed with importance weights.

`conditional(`*if*`)` specifies that the imputation variable be imputed conditionally on observations satisfying *exp*; see [U] **11.1.3 if exp**. That is, missing values in a conditional sample, the sample identified by the *exp* expression, are imputed based only on data in that conditional sample. Missing values outside the conditional sample are replaced with a conditional constant, the value of the imputation variable in observations outside the conditional sample. As such, the imputation variable is required to be constant outside the conditional sample. Also, if any conditioning variables (variables involved in the conditional specification `if` *exp*) contain soft missing values (`.`), their missing values must be nested within missing values of the imputation variables. See *Conditional imputation* under *Remarks and examples* in [MI] **mi impute**.

`bootstrap` specifies that posterior estimates of model parameters be obtained using sampling with replacement; that is, posterior estimates are estimated from a bootstrap sample. The default is to sample the estimates from the posterior distribution of model parameters or from the large-sample normal approximation of the posterior distribution. This option is useful when asymptotic normality of parameter estimates is suspect.

Reporting

`dots`, `noisily`, `nolegend`; see [MI] **mi impute**. `noisily` specifies that the output from the logistic regression fit to the observed data be displayed. `nolegend` suppresses all legends that appear before

the imputation table. Such legends include a legend about conditional imputation that appears when the `conditional()` option is specified and group legends that may appear when the `by()` option is specified.

Maximization

maximize_options; see [R] **logit**. These options are seldom used. `difficult`, `technique()`, `gradient`, `showstep`, `hessian`, and `showtolerance` are not allowed when the `augment` option is used.

Advanced

`force`; see [MI] **mi impute**.

The following option is available with `mi impute` but is not shown in the dialog box:

`noupdate`; see [MI] **noupdate option**.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Univariate imputation using logistic regression
Using mi impute logit
Video example

See [MI] **mi impute** for a general description and details about options common to all imputation methods, *impute_options*. Also see [MI] **Workflow** for general advice on working with `mi`.

Univariate imputation using logistic regression

The logistic regression imputation method can be used to fill in missing values of a binary variable (for example, Rubin [1987]; Raghunathan et al. [2001]; and van Buuren [2007]). It is a parametric method that assumes an underlying logistic model for the imputed variable (given other predictors).

Unlike the linear regression method, the logistic imputation method is based on the asymptotic approximation of the posterior predictive distribution of the missing data. The actual posterior distribution of the logistic model parameters, β, does not have a simple form under the common noninformative prior distribution. Thus a large-sample normal approximation to the posterior distribution of β is used instead. Rubin (1987, 169) points out that although the actual posterior distribution may be far from normal (for example, when the number of observed cases is small or when the fraction of ones in the observed data is close to zero or one), the use of the normal approximation is common in practice.

Using mi impute logit

Continuing our heart attack example from [MI] **Intro substantive** and [MI] **mi impute**, suppose that `hsgrad`, a binary variable recording whether subjects graduated from high school, contains missing values:

```
. use https://www.stata-press.com/data/r17/mheart2
(Fictional heart attack data; hsgrad missing)
. mi set mlong
. mi misstable summarize
```

				Obs<.		
Variable	Obs=.	Obs>.	Obs<.	Unique values	Min	Max
hsgrad	18		136	2	0	1

Thus we want to impute missing values of `hsgrad`, because `hsgrad` was one of the predictors in our logistic model (`logit attack smokes age bmi female hsgrad`). From our previous analysis of the heart attack data, we recall that `hsgrad` was not a significant predictor. So, we could have omitted `hsgrad` from the logistic model in the casewise-deletion analysis to avoid the reduction in sample size, and then imputing `hsgrad` would not have been needed. In general, the imputer rarely has such knowledge, and omitting `hsgrad` from the imputation model would prevent this predictor from being used in later analysis by the analyst (see, for example, *Imputation modeling* in [MI] **mi impute**). Thus we proceed with imputation.

We use `mi impute logit` to create 10 imputations of `hsgrad`:

```
. mi register imputed hsgrad
(18 m=0 obs now marked as incomplete)
. mi impute logit hsgrad attack smokes age bmi female, add(10)
Univariate imputation                       Imputations =       10
Logistic regression                                added =       10
Imputed: m=1 through m=10                        updated =        0
```

	Observations per *m*			
Variable	Complete	Incomplete	Imputed	Total
hsgrad	136	18	18	154

```
(Complete + Incomplete = Total; Imputed is the minimum across m
 of the number of filled-in observations.)
```

We can now use the imputed `hsgrad` in our analysis, for example,

```
. mi estimate: logit attack smokes age bmi female hsgrad
  (output omitted)
```

Video example

Multiple imputation: Setup, imputation, estimation—logistic regression

Stored results

`mi impute logit` stores the following in `r()`:

Scalars

`r(M)`	total number of imputations
`r(M_add)`	number of added imputations
`r(M_update)`	number of updated imputations
`r(k_ivars)`	number of imputed variables (always `1`)
`r(pp)`	`1` if perfect prediction detected, `0` otherwise
`r(N_g)`	number of imputed groups (`1` if `by()` is not specified)

Macros

`r(method)`	name of imputation method (`logit`)
`r(ivars)`	names of imputation variables
`r(rngstate)`	random-number state used
`r(by)`	names of variables specified within `by()`

Matrices

`r(N)`	number of observations in imputation sample in each group
`r(N_complete)`	number of complete observations in imputation sample in each group
`r(N_incomplete)`	number of incomplete observations in imputation sample in each group
`r(N_imputed)`	number of imputed observations in imputation sample in each group

Methods and formulas

Consider a univariate variable $\mathbf{x} = (x_1, x_2, \ldots, x_n)'$ that follows a logistic model

$$\Pr(x_i \neq 0|\mathbf{z}_i) = \frac{\exp(\mathbf{z}_i'\boldsymbol{\beta})}{1 + \exp(\mathbf{z}_i'\boldsymbol{\beta})} \tag{1}$$

where $\mathbf{z}_i = (z_{i1}, z_{i2}, \ldots, z_{iq})'$ records values of predictors of $\mathbf{x}$ for observation i and $\boldsymbol{\beta}$ is the $q \times 1$ vector of unknown regression coefficients. (When a constant is included in the model—the default—$z_{i1} = 1,\ i = 1, \ldots, n$.)

$\mathbf{x}$ contains missing values that are to be filled in. Consider the partition of $\mathbf{x} = (\mathbf{x}_o', \mathbf{x}_m')$ into $n_0 \times 1$ and $n_1 \times 1$ vectors containing the complete and the incomplete observations. Consider a similar partition of $\mathbf{Z} = (\mathbf{Z}_o, \mathbf{Z}_m)$ into $n_0 \times q$ and $n_1 \times q$ submatrices.

`mi impute logit` follows the steps below to fill in $\mathbf{x}_m$:

1. Fit a logistic model (1) to the observed data $(\mathbf{x}_o, \mathbf{Z}_o)$ to obtain the maximum likelihood estimates, $\widehat{\boldsymbol{\beta}}$, and their asymptotic sampling variance, $\widehat{\mathbf{U}}$.
2. Simulate new parameters, $\boldsymbol{\beta}_\star$, from the large-sample normal approximation, $N(\widehat{\boldsymbol{\beta}}, \widehat{\mathbf{U}})$, to its posterior distribution assuming the noninformative prior $\Pr(\boldsymbol{\beta}) \propto \text{const}$.
3. Obtain one set of imputed values, $\mathbf{x}_m^1$, by simulating from the logistic distribution:

$$\Pr(x_{i_m} = 1) = \exp(\mathbf{z}_{i_m}'\boldsymbol{\beta}_\star) / \left\{1 + \exp(\mathbf{z}_{i_m}'\boldsymbol{\beta}_\star)\right\}$$

   for every missing observation i_m.
4. Repeat steps 2 and 3 to obtain M sets of imputed values, $\mathbf{x}_m^1, \mathbf{x}_m^2, \ldots, \mathbf{x}_m^M$.

Steps 2 and 3 above correspond to only approximate draws from the posterior predictive distribution of the missing data $\Pr(\mathbf{x}_m|\mathbf{x}_o, \mathbf{Z}_o)$ because $\boldsymbol{\beta}_\star$ is drawn from the asymptotic approximation to its posterior distribution.

If weights are specified, a weighted logistic regression model is fit to the observed data in step 1 (see [R] **logit** for details).

References

Raghunathan, T. E., J. M. Lepkowski, J. Van Hoewyk, and P. Solenberger. 2001. A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey Methodology* 27: 85–95.

Rubin, D. B. 1987. *Multiple Imputation for Nonresponse in Surveys*. New York: Wiley.

van Buuren, S. 2007. Multiple imputation of discrete and continuous data by fully conditional specification. *Statistical Methods in Medical Research* 16: 219–242. https://doi.org/10.1177/0962280206074463.

Also see

[MI] **mi impute** — Impute missing values

[MI] **mi estimate** — Estimation using multiple imputations

[MI] **Intro** — Introduction to mi

[MI] **Intro substantive** — Introduction to multiple-imputation analysis