Package: mlim (via r-universe)

October 19, 2026

Type Package

Title Single and Multiple Imputation with Automated Machine Learning

Version 0.6.0

Depends R (>= 3.5.0)

Description Machine learning algorithms have been used for performing single missing data imputation and most recently, multiple imputations. However, this is the first attempt for using automated machine learning algorithms for performing both single and multiple imputation. Automated machine learning is a procedure for fine-tuning the model automatic, performing a random search for a model that results in less error, without overfitting the data. The main idea is to allow the model to set its own parameters for imputing each variable separately instead of setting fixed predefined parameters to impute all variables of the dataset. Using automated machine learning, the package fine-tunes an Elastic Net (default) or Gradient Boosting, Random Forest, Deep Learning, Extreme Gradient Boosting, or Stacked Ensemble machine learning model (from one or a combination of other supported algorithms) for imputing the missing observations. This procedure has been implemented for the first time by this package and is expected to outperform other packages for imputing missing data that do not fine-tune their models. The multiple imputation is implemented via bootstrapping without letting the duplicated observations to harm the cross-validation procedure, which is the way imputed variables are evaluated. Most notably, the package implements automated procedure for handling imputing imbalanced data (class rarity problem), which happens when a factor variable has a level that is far more prevalent than the other(s). This is known to result in biased predictions, hence, biased imputation of missing data. However, the autobalancing procedure ensures that instead of focusing on maximizing accuracy (classification error) in imputing factor variables, a fairer procedure and imputation method is practiced.

License MIT + file LICENSE

Encoding UTF-8

Imports mice, missRanger, memuse, mlr3, mlr3pipelines, mlr3tuning, paradox, md.log (>= 0.2.0), readstata13 (>= 0.11.0)

Suggests mlr3learners

LazyData true

URL https://github.com/haghish/mlim

BugReports https://github.com/haghish/mlim/issues

Config/roxygen2/version 8.1.0

Config/pak/sysreqs cmake make libicu-dev libx11-dev zlib1g-dev

Repository https://haghish.r-universe.dev

Date/Publication 2026-10-19 18:43:02 UTC

RemoteUrl https://github.com/haghish/mlim

RemoteRef HEAD

RemoteSha ee37292c12648eb35095ba6f7d425290041b97c8

Contents

charity 2
egsingle 3
manifest 4
mlim 7
mlim.error 10
mlim.mids 12
mlim.na 13
mlim.na.multilevel 14
mlim.preimpute 16
mlim.stata 17
mlim.summarize 18

Index **20**

`charity` *Attitudes toward charitable organizations*

Description

A dataset containing five Likert-scale items assessing attitudes toward charitable organizations, including perceived effectiveness, trust, honesty and ethical conduct, community impact, and service delivery.

Usage

```
charity
```

Format

A data frame with 832 observations and 5 variables:

ta1 Perceived effectiveness of charitable organizations.

ta2 Degree of trust in charitable organizations.

ta3 Perceived honesty and ethical conduct of charitable organizations.

ta4 Perceived role of charitable organizations in improving communities.

ta5 Perceived effectiveness of charitable organizations in delivering services.

Source

https://www.stata.com/

---

egsingle *US Sustaining Effects Study*

Description

A subset of the mathematics scores from the U.S. Sustaining Effects Study. The dataset contains repeated observations for 1,721 students from 60 schools.

Usage

```
data(egsingle)
```

Format

A data frame with 7,230 observations and 12 variables:

schoolid A factor identifying schools.

childid A factor identifying students.

year A numeric variable indicating the year of the test.

grade A numeric variable indicating the student's grade.

math A numeric variable containing mathematics test scores on the IRT scale-score metric.

retained A factor with levels `"0"` and `"1"` indicating whether the student has been retained in a grade.

female A factor with levels `"Female"` and `"Male"` indicating the student's sex.

black A factor with levels `"0"` and `"1"` indicating whether the student is Black.

hispanic A factor with levels `"0"` and `"1"` indicating whether the student is Hispanic.

size A numeric variable indicating the number of students enrolled in the school.

lowinc A numeric variable giving the percentage of low-income students in the school.

mobility A numeric variable measuring student mobility.

Details

This dataset is borrowed from the **mlmRev** R package.

Source

The dataset is borrowed from the **mlmRev** R package. The original data were distributed with the HLM software package (Bryk, Raudenbush, and Congdon, 1996). Conversion to R format is described in Doran and Lockwood (2004).

References

Doran, H. C., & Lockwood, J. R. (2004). *Fitting value-added models in R*.

Examples

```
data(egsingle)
str(egsingle)

## Not run:
library(lme4)
fm1 <- lmer(
  math ~ year * size + female + (1 | childid) + (1 | schoolid),
  data = egsingle
)
summary(fm1)

## End(Not run)
```

`manifest` *Manifest Anxiety Scale*

Description

The Taylor Manifest Anxiety Scale was first developed in 1953 to identify individuals who would be good subjects for studies of stress and other related psychological phenomenon. Since then it has been used as a measure of anxiety as general personality trait. Anxiety is a complex psychological construct that includes a multiple of different facets related to extensive worrying that may impair normal functioning. The test has been widely studied and used in research, however there are some concerns that it does not measure a single trait, but instead, measures a basket of loosely related ones and so the score is not that meaningful.

Usage

```
manifest
```

Format

A data frame with 4469 rows and 52 variables:

gender participants' gender

age participants' age in years

Q1 I do not tire quickly.

Q2 I am troubled by attacks of nausea.

Q3 I believe I am no more nervous than most others.

Q4 I have very few headaches.

Q5 I work under a great deal of tension.

Q6 I cannot keep my mind on one thing.

Q7 I worry over money and business.

Q8 I frequently notice my hand shakes when I try to do something.

Q9 I blush no more often than others.

Q10 I have diarrhea once a month or more.

Q11 I worry quite a bit over possible misfortunes.

Q12 I practically never blush.

Q13 I am often afraid that I am going to blush.

Q14 I have nightmares every few nights.

Q15 My hands and feet are usually warm.

Q16 I sweat very easily even on cool days.

Q17 Sometimes when embarrassed, I break out in a sweat.

Q18 I hardly ever notice my heart pounding and I am seldom short of breath.

Q19 I feel hungry almost all the time.

Q20 I am very seldom troubled by constipation.

Q21 I have a great deal of stomach trouble.

Q22 I have had periods in which I lost sleep over worry.

Q23 My sleep is fitful and disturbed.

Q24 I dream frequently about things that are best kept to myself.

Q25 I am easily embarrassed.

Q26 I am more sensitive than most other people.

Q27 I frequently find myself worrying about something.

Q28 I wish I could be as happy as others seem to be.

Q29 I am usually calm and not easily upset.

Q30 I cry easily.

Q31 I feel anxiety about something or someone almost all the time.

Q32 I am happy most of the time.

Q33 It makes me nervous to have to wait.

Q34 I have periods of such great restlessness that I cannot sit long I a chair.

Q35 Sometimes I become so excited that I find it hard to get to sleep.

Q36 I have sometimes felt that difficulties were piling up so high that I could not overcome them.

Q37 I must admit that I have at times been worried beyond reason over something that really did not matter.

Q38 I have very few fears compared to my friends.

Q39 I have been afraid of things or people that I know could not hurt me.

Q40 I certainly feel useless at times.

Q41 I find it hard to keep my mind on a task or job.

Q42 I am usually self-conscious.

Q43 I am inclined to take things hard.

Q44 I am a high-strung person.

Q45 Life is a trial for me much of the time.

Q46 At times I think I am no good at all.

Q47 I am certainly lacking in self-confidence.

Q48 I sometimes feel that I am about to go to pieces.

Q49 I shrink from facing crisis of difficulty.

Q50 I am entirely self-confident.

Details

The data comes from an online offering of the Taylor Manifest Anxiety Scale. At the end of the test users were asked if their answers were accurate and could be used for research, 76 https://openpsychometrics.org/.

items 1 to 50 were rated 1=True and 2=False. gender, chosen from a drop down menu (1=male, 2=female, 3=other) and age was entered as a free response (ages<14 have been removed)

Source

https://openpsychometrics.org/tests/TMAS/

References

Taylor, J. (1953). "A personality scale of manifest anxiety". The Journal of Abnormal and Social Psychology, 48(2), 285-290.

mlim *missing data imputation with automated machine learning*

Description

imputes data.frame with mixed variable types using automated machine learning (AutoML)

Usage

```
mlim(
  data = NULL,
  m = 1,
  algos = c("ELNET", "XGB"),
  preimpute = "random",
  ignore = NULL,
  hierarchy = NULL,
  tolerance = 0.001,
  tuning_time = 300,
  max_models = 50,
  maxiter = 15L,
  cv = 5L,
  cpu = 1,
  stochastic = m > 1,
  autobalance = TRUE,
  seed = NULL,
  verbosity = NULL,
  report = NULL,
  save = NULL,
  load = NULL,
  ...
)
```

Arguments

`data` a `data.frame` (strictly) with missing data to be imputed. if `'load'` argument is provided, this argument will be ignored.

`m` integer, specifying number of multiple imputations. the default value is 1, carrying out a single imputation.

`algos` character vector specifying the machine-learning algorithms used for imputation. Supported algorithms are `"ELNET"` (elastic net via `glmnet`), `"RF"` (random forest via `ranger`), `"CRF"` (conditional random forest via `partykit::cforest`), `"GBM"` (classical gradient boosting via `gbm`), `"XGB"` (XGBoost), `"LGBM"` (LightGBM), `"CAT"` (CatBoost), `"NNET"` (single-hidden-layer neural network via `nnet`), `"SVM"` (kernel support vector machine via `kernlab::ksvm`), `"KNN"` (k-nearest neighbors), and `"ENSEMBLE"`. The default is `"ELNET"`.

When several base algorithms are supplied, `max_models` and `tuning_time` are divided across the base algorithms and the best-performing candidate is used.

If "ENSEMBLE" is included, at least two additional base algorithms must also be supplied. The ensemble is a stacked model constructed with mlr3pipelines using cross-validated predictions from the successfully tuned base learners and is evaluated as an additional candidate after base-learner tuning.

The current mlr3extralearners classif.gbm wrapper supports two-class classification but not multiclass classification; therefore "GBM" is skipped for multinomial targets when other learners are available.

"KNN" does not support observation weights in its current mlr3 learner. It can therefore be evaluated in single imputation, but is skipped during multiple imputation because bootstrap multiplicity weights are required for model fitting. When class balancing is requested in single imputation, KNN is fitted without learner weights, although balancing weights are retained for performance assessment.

preimpute — Character specifying the initial treatment of missing values before iterative model-based imputation. The default is "random", which performs random sampling from each feature. The alternative is "mm", which performs median/mode preimputation.

ignore — character vector of column names or index of columns that should should be ignored in the process of imputation.

hierarchy — Character vector specifying the clustering variables from the highest to the lowest level. For example, hierarchy = c("city", "school", "classroom", "student") specifies students nested within classrooms, classrooms nested within schools, and schools nested within cities. Hierarchy variables must exist in data and cannot contain missing values. The default is NULL, which assumes no hierarchical structure.

tolerance — numeric. the minimum rate of improvement in estimated error metric of a variable to qualify the imputation for another round of iteration, if the maxiter is not yet reached. any improvement of imputation is desirable. however, specifying values above 0 can reduce the number of required iterations at a marginal increase of imputation error. for larger datasets, value of "1e-3" is recommended to reduce number of iterations. the default value is '1e-3'.

tuning_time — Numeric. Maximum base-learner tuning runtime in seconds for each variable and iteration. The default is 3600. When several base algorithms are selected, this budget is divided across them. Tuning stops when the applicable time or evaluation limit is reached. A requested stacked ensemble is evaluated after base-learner tuning and does not consume this base-learner tuning-time allocation.

max_models — Integer or NULL. Maximum number of hyperparameter evaluations across the base algorithms for each variable and iteration. The default is 100. When several base algorithms are selected, this budget is divided across them. If NULL, no explicit evaluation-count limit is supplied by mlim. "ENSEMBLE" does not count as a base algorithm for this allocation.

maxiter — integer. maximum number of iterations. the default value is 15, but it can be reduced to 3 (not recommended, see below).

cv — Integer specifying the number of cross-validation folds. Values of 5 or higher are required. the default is 5.

`cpu` Integer specifying the number of CPU threads supplied to learners that support internal multithreading. The default is `1`.

`stochastic` Logical. If `TRUE`, stochastic variation is added after each accepted variable-specific imputation update. For continuous variables, values are drawn from a normal distribution centered on the model prediction with the current cross-validation RMSE as the standard deviation. For categorical variables, values are sampled from the predicted class probabilities. The default is `FALSE` for single imputation and `TRUE` for multiple imputation.

`autobalance` logical. if TRUE (default), binary and multinomial factor variables are balanced during single imputation. During multiple imputation, balancing weights are combined with bootstrap multiplicity weights.

`seed` integer. specify the random generator seed

`verbosity` character. controls how much information is printed to console. the value can be "warn" (default), "info", "debug", or NULL.

`report` filename. if a filename is specified (e.g. report = "mlim.md"), the `"md.log"` R package is used to generate a Markdown progress report for the imputation. the format of the report is adopted based on the `'verbosity'` argument. the higher the verbosity, the more technical the report becomes. if verbosity equals "debug", then a log file is generated, which includes time stamp and shows the function that has generated the message. otherwise, a reduced markdown-like report is generated. default is NULL.

`save` filename (with .mlim extension). if a filename is specified, an `mlim` object is saved after the end of each variable imputation. this object not only includes the imputed dataframe and estimated cross-validation error, but also includes the information needed for continuing the imputation, which is very useful feature for imputing large datasets, with a long runtime. this argument is activated by default and an mlim object is stored in the local directory named `"mlim.rds"`.

`load` filename (with .mlim extension). an object of class "mlim", which includes the data, arguments, and settings for re-running the imputation, from where it was previously stopped. the "mlim" object saves the current state of the imputation and is particularly recommended for large datasets or when the user specifies a computationally extensive settings (e.g. specifying several algorithms, increasing tuning time, etc.).

`...` arguments that are used internally between 'mlim' these arguments are not documented in the help file and are not intended to be used by end user.

Value

a `data.frame`, showing the estimated imputation error from the cross validation within the data.frame's attribution

Author(s)

E. F. Haghish

Examples

```
## Not run:
data(iris)

# add stratified missing observations to the data. to make the example run
# faster, I add NAs only to a single variable.
dfNA <- iris
dfNA$Species <- mlim.na(dfNA$Species, p = 0.1, stratify = TRUE, seed = 2022)

# run the ELNET single imputation (fastest imputation via 'mlim')
MLIM <- mlim(dfNA)

# in single imputation, you can estimate the imputation accuracy via cross validation RMSE
mlim.summarize(MLIM)

### or if you want to carry out ELNET multiple imputation with 5 datasets.
### next, to carry out analysis on the multiple imputation, use the 'mlim.mids' function
### minimum of 5 datasets
MLIM2 <- mlim(dfNA, m = 5)
mids <- mlim.mids(MLIM2, dfNA)
fit <- with(data=mids, exp=glm(Species ~ Sepal.Length, family = "binomial"))
res <- mice::pool(fit)
summary(res)

# you can check the accuracy of the imputation, if you have the original dataset
mlim.error(MLIM2, dfNA, iris)

## End(Not run)
```

`mlim.error` *Evaluate imputation error*

Description

Calculates normalized RMSE for numeric variables, mean per-class error for unordered factors, and normalized rank error for ordered factors.

Usage

```
mlim.error(
  imputed,
  incomplete,
  complete,
  transform = NULL,
  varwise = FALSE,
  ignore.missclass = TRUE,
  ignore.rank = FALSE
)
```

Arguments

`imputed`	Imputed data frame, `mlim` object, `mlim.mi` object, list of completed data frames, or `mids` object.
`incomplete`	Original incomplete data frame.
`complete`	Original complete data frame used as the reference.
`transform`	Optional transformation for numeric variables. Supported values are `"standardize"` and `"normalize"`. The same transformation, estimated from `complete`, is applied to all three datasets.
`varwise`	Logical. If `TRUE`, variable-wise error estimates are returned in addition to overall estimates.
`ignore.missclass`	Logical. If `FALSE`, ordinary misclassification error is also returned for unordered factors. The default is `TRUE`.
`ignore.rank`	Logical. If `FALSE`, ordered factors are evaluated using normalized rank distance. If `TRUE`, they are treated as unordered factors.

Value

A named numeric vector, or a list when `varwise = TRUE`. For multiple imputations, returns a matrix or a list of variable-wise results.

Author(s)

E. F. Haghish

Examples

```
## Not run:
data(iris)
irisNA <- mlim.na(
  iris, p = 0.1,
  stratify = TRUE,
  seed = 2022
)

imp <- mlim(irisNA)
mlim.error(imp, irisNA, iris)

mlim.error(
  imp, irisNA, iris,
  varwise = TRUE
)

## End(Not run)
```

`mlim.mids` *convert multiple imputations to a mids object*

Description

converts multiply imputed datasets to a `mids` object for analysis with `mice`.

Usage

```
mlim.mids(mlim, incomplete)
```

Arguments

`mlim`	An object of class `"mlim.mi"` returned by `mlim()`, or a compatible MI object.
`incomplete`	The original incomplete data frame.

Details

The original data are stored as imputation 0 and completed datasets as imputations 1, ..., m. The function verifies dataset dimensions, variable order, observed values, and completion of originally missing values before conversion.

Value

An object of class `"mids"`.

Author(s)

E. F. Haghish

Examples

```
## Not run:
data(iris)
irisNA <- mlim.na(iris, p = 0.1, seed = 2022)
imp <- mlim(
  irisNA, m = 5, tuning_time = 180, seed = 2022
)
mids <- mlim.mids(imp, irisNA)
fit <- with(
  mids,
  lm(Sepal.Length ~ Sepal.Width + Petal.Length)
)
summary(mice::pool(fit))

## End(Not run)
```

`mlim.na` *add stratified/unstratified artificial missing observations*

Description

to examine the performance of imputation algorithms, artificial missing data are added to datasets and then imputed, to compare the original observations with the imputed values. this function can add stratified or unstratified artificial missing data. stratified missing data can be particularly useful if your categorical or ordinal variables are imbalanced, i.e., one category appears at a much higher rate than others.

Usage

```
mlim.na(x, p = 0.1, stratify = FALSE, classes = NULL, seed = NULL)
```

Arguments

`x`	data.frame. x must be strictly a data.frame and any other data.table classes will be rejected
`p`	proportion of missingness to be added to the data
`stratify`	logical. if TRUE, stratified sampling will be carried out, when adding NA values to 'factor' variables (either ordered or unordered). this feature makes evaluation of missing data imputation algorithms more fair, especially when the factor levels are imbalanced.
`classes`	character vector, specifying the variable classes that should be selected for adding NA values. the default value is NULL, meaning all variables will receive NA values with probability of 'p'. however, if you wish to add NA values only to a specific classes, e.g. 'numeric' variables or 'ordered' factors, specify them in this argument. e.g. write "classes = c('numeric', 'ordered')" if you wish to add NAs only to numeric and ordered factors.
`seed`	integer. a random seed number for reproducing the result (recommended)

Value

data.frame

Author(s)

E. F. Haghish

Examples

```
## Not run:
# adding stratified NA to an atomic vector
x <- as.factor(c(rep("M", 100), rep("F", 900)))
table(mlim.na(x, p=.5, stratify = TRUE))
```

```
# adding unstratified NAs to all variables of a data.frame
data(iris)
mlim.na(iris, p=0.5, stratify = FALSE, seed = 1)

# or add stratified NAs only to factor variables, ignoring other variables
mlim.na(iris, p=0.5, stratify = TRUE, classes = "factor", seed = 1)

# or add NAs to numeric variables
mlim.na(iris, p=0.5, classes = "numeric", seed = 1)

## End(Not run)
```

`mlim.na.multilevel` *add artificial missing observations at a hierarchical level*

Description

to examine the performance of imputation algorithms in hierarchical data, artificial missing data can be added at a selected level of the hierarchy. instead of sampling individual rows, this function samples complete clusters and sets the selected variables to NA for all observations belonging to the sampled clusters. because complete clusters are removed, the requested missing proportion is treated as a target and may not be achieved exactly.

Usage

```
mlim.na.multilevel(x, p = 0.1, hierarchy, level, variables = NULL, seed = NULL)
```

Arguments

`x`	data.frame. x must be strictly a data.frame and any other data.table classes will be rejected
`p`	target proportion of currently observed values to be replaced by NA in the selected variables. because complete clusters are sampled, the achieved proportion can differ slightly from 'p'.
`hierarchy`	character vector specifying the clustering variables from the highest to the lowest level. for example, 'hierarchy = c("schoolid", "childid")' specifies children nested within schools.
`level`	character string specifying the hierarchical level at which missingness should be generated. 'level' must name one variable in 'hierarchy'. for example, 'level = "childid"' samples complete children, whereas 'level = "schoolid"' samples complete schools.
`variables`	character vector specifying the variables that should receive NA values. the default value is NULL, meaning all variables not included in 'hierarchy' are selected. the hierarchy variables themselves cannot receive missing values.
`seed`	integer. a random seed number for reproducing the result (recommended)

Details

nesting is defined cumulatively. for example, with 'hierarchy = c("schoolid", "childid")' and 'level = "childid"', children are identified by the combination of 'schoolid' and 'childid'. this allows child identifiers to be repeated across schools.

the function first counts the currently observed cells in the selected variables for each eligible cluster. clusters are then randomly ordered, and a complete cluster is selected whenever adding it moves the achieved amount of missingness closer to the target defined by 'p'. consequently, 'p' represents a target proportion of added missingness rather than an exact cluster sampling fraction.

when several variables are supplied in 'variables', the same sampled clusters receive missing values for all selected variables. to create independent hierarchical missingness patterns for different variables, call the function separately for each variable or variable set.

Value

data.frame. the returned data frame contains the added missing values. the target proportion, achieved proportion, selected hierarchy level, selected variables, and sampled clusters are stored as attributes 'mlim.na.target.p', 'mlim.na.achieved.p', 'mlim.na.level', 'mlim.na.variables', and 'mlim.na.selected.clusters', respectively. 'mlim.na.selected.clusters' is a data frame containing the hierarchy values that identify each sampled cluster.

Author(s)

E. F. Haghish

Examples

```
## Not run:
if (requireNamespace("mlmRev", quietly = TRUE)) {

  data("egsingle", package = "mlmRev")

  dat <- egsingle

  # Add child-level missingness to female for approximately 20 percent
  # of the currently observed values
  dat_child <- mlim.na.multilevel(
    dat,
    p = 0.20,
    hierarchy = c("schoolid", "childid"),
    level = "childid",
    variables = "female",
    seed = 2022
  )

  # Add school-level missingness to black and hispanic. The same schools
  # will have both variables set to NA.
  dat_school <- mlim.na.multilevel(
    dat,
    p = 0.20,
    hierarchy = c("schoolid", "childid"),
```

```
    level = "schoolid",
    variables = c("black", "hispanic"),
    seed = 2022
  )

  # Inspect the achieved proportion of added missingness
  attr(dat_school, "mlim.na.achieved.p")
}

## End(Not run)
```

`mlim.preimpute` *Preimputation of missing values*

Description

Initializes missing values before the iterative mlim imputation procedure. Missing values can be initialized using median/mode, random sampling, from the observed values of each variable, or Random Forest imputation (experimental).

Usage

```
mlim.preimpute(data, preimpute = "random", seed = NULL)
```

Arguments

`data`	data.frame containing missing values
`preimpute`	character. Specify the algorithm for preimputation. Supported options are "mm" (median/mode replacement), "random" for random sampling from available data.
`seed`	integer. Random-number seed used by Random Forest or random sampling. The default is `NULL`.

Details

NOTE: ====

This file includes "RF" for randomforest preimputation. Such an approach can inflate relationships between the features and thus, this method is not documented and is not recommended. It is maintained in the code for research purpose only.

Value

imputed data.frame

Author(s)

E. F. Haghish

Examples

```
## Not run:
data(iris)

# add 10% stratified missing values to one factor variable
irisNA <- iris
irisNA$Species <- mlim.na(irisNA$Species, p = 0.1, stratify = TRUE, seed = 2022)

# run the default Median/Model preimputation
MLIM <- mlim.preimpute(irisNA, preimpute = "mm")
mlim.error(MLIM, irisNA, iris) #check the preimputation error

# Random-sampling preimputation
RANDOM <- mlim.preimpute(irisNA, preimpute = "random", seed = 2022)

## End(Not run)
```

`mlim.stata` *Prepare Multiple Imputation Data for Stata*

Description

Converts an object of class `mlim.mi` to a data frame that can be imported into Stata as multiple imputation data. Currently, only Stata's `flong` format is supported.

Usage

```
mlim.stata(mlim, df, format = "flong", filename = NULL)
```

Arguments

`mlim`	An object of class `mlim.mi` containing the multiple imputation datasets.
`df`	A data frame containing the original unimputed dataset used for imputation.
`format`	Character string specifying the Stata multiple imputation format. Currently, only `"flong"` is supported.
`filename`	Optional character string specifying the file name or path for saving the prepared dataset as a Stata `.dta` file.

Details

For the `flong` format, the original unimputed dataset is assigned `m = 0`, and the imputed datasets are assigned `m = 1, ..., M`. An `id` variable is added to identify the same observation across the original and imputed datasets. All datasets are then apprended by rows.

Value

A data frame containing the original and imputed datasets in Stata `flong` format.

Examples

```
## Not run:
imp <- mlim(data = df, m = 5)

stata.data <- mlim.stata(
  mlim = imp,
  df = df,
  format = "flong"
)

mlim.stata(
  mlim = imp,
  df = df,
  filename = "imputed_data.dta"
)

## End(Not run)
```

`mlim.summarize` *Summarize mlim imputation error*

Description

Summarizes the best accepted cross-validation RMSE for each imputed variable.

Usage

```
mlim.summarize(data)
```

Arguments

`data`	An object returned by `mlim()`.

Value

For a single imputation, a data frame containing variable names and RMSE values. For multiple imputation, a data frame containing one RMSE column per imputation.

Author(s)

E. F. Haghish

Examples

```
## Not run:
data(iris)

irisNA <- iris
irisNA$Species <- mlim.na(
  irisNA$Species,
  p = 0.1,
  stratify = TRUE,
  seed = 2022
)

imp <- mlim(irisNA)
mlim.summarize(imp)

## End(Not run)
```

Index

∗ **datasets**
    charity, 2
    egsingle, 3
    manifest, 4

charity, 2

egsingle, 3

manifest, 4
mlim, 7
mlim.error, 10
mlim.mids, 12
mlim.na, 13
mlim.na.multilevel, 14
mlim.preimpute, 16
mlim.stata, 17
mlim.summarize, 18