Package 'SURF'

November 3, 2017

Type Package

Title Some Useful R Functions

Version 0.0.0.9000

Date 2017-11-03

Author Kyle M. Lang

Maintainer Kyle M. Lang <k.m.lang@uvt.nl>

Description These are several useful functions that I find myself using often (read: frequently re-implementing), so I will package them for easy access/dissemination.

License GPL-3 | file LICENSE

Depends mvtnorm

R topics documented:

SURF-package *Some Useful R Functions*

Description

These are several useful functions that I find myself using often (read: frequently re-implementing), so I will package them for easy access/dissemination.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

Author(s)

Kyle M. Lang

Maintainer: Kyle M. Lang <k.m.lang@uvt.nl>

Examples

```
## Simulate regression data:
testData <- simRegData(nObs   = 100,
                       nPreds = 10,
                       r2     = 0.5,
                       collin = 0.2,
                       beta   = matrix(c(0.25, rep(0.75, 10)))
                       )

## Impose missing data
missData <- imposeMissData(data     = testData,
                           targets  = list(mar  = c("y", "x1"),
                                           mcar = c("x2", "x3"),
                                           mnar = c("x4", "x5")
                                           ),
                            preds   = c("x8", "x9", "x10"),
                            pm      = list(mar = 0.2, mcar = 0.1, mnar = 0.1),
                            snr     = list(mar = 5, mnar = 2.5),
                           pattern = "random")

## Plot imputed vs. observed values:
data(testImps)

plotImps(impList = testImps$impList,
         rMat    = testImps$rMat,
         typeVec = testImps$typeVec)
```

---

imposeMissData *Impose Missing Data*

Description

Impose missing data according to MAR, MCAR, and MNAR mechanisms.

Usage

```
imposeMissData(data,
               targets,
               preds,
               pm,
               snr,
               pattern = "random")
```

Arguments

`data`	A data.frame wherein missing data are to be imposed.
`targets`	A named list with slots "mar", "mcar", and "mnar" containing character vectors giving the column labels for variables onto which missing at random, missing completely at random, and missing not at random data, respectively, shall be imposed.
`preds`	A character vector giving the column labels for predictors of the MAR missingness.
`pm`	A named list with slots "mar", "mcar", and "mnar" containing real numbers in [0, 1) giving the proportions of missing at random, missing completely at random, and missing not at random data, respectively, to generate.
`snr`	A named list with slots "mar" and "mnar" containing real numbers giving the signal-to-noise ratios of the probit regression models used to impose missing at random and missing not at random data, respectively.
`pattern`	A character vector indicating in what parts of the missing data predictors' distributions MAR and MNAR missing data should be imposed. Legal keywords are: "low" = impose missing in the negative tail of the predictor, "high" = impose missing in the positive tail of the predictor, "center" = impose missing in the center of the predictor, "tails" = impose missing in both tails of the predictor. The `pattern` arugment can also be the special keyword "random" which will cause the function to randomly sample from the four preceding possiblities for each target variable.

Details

MCAR missing data is imposed by generating a random Bernoulli flag variable for each target variable with probablity of success equal to `pm$mcar`.

MAR missing data is imposed via a noisy probit regression model wherein the weighted sum of the columns listed in `preds` are used to predict the response propensity.

MNAR missing data is imposed via the same procedure as MAR missing data but the missing data predictor is taken to be the target variable itself.

Value

A two-element list with the following slots:

data: The incomplete version of `data`

pattern: A character vector showing which `pattern` was used for each target varaible.

Note

Due to the stochastic nature of the missing data simulation, the actual proportions of missing data will only equal the values provided for `pm` asymptotically.

Author(s)

Kyle M. Lang

See Also

simRegData

Examples

```
## Simulate some data:
testData <- simRegData(nObs   = 100,
                       nPreds = 10,
                       r2     = 0.5,
                       collin = 0.2,
                       beta   = matrix(c(0.25, rep(0.75, 10)))
                       )

## Impose missing data:
missData <- imposeMissData(data     = testData,
                           targets  = list(mar  = c("y", "x1"),
                                           mcar = c("x2", "x3"),
                                           mnar = c("x4", "x5")
                                           ),
                            preds   = c("x8", "x9", "x10"),
                            pm      = list(mar = 0.2, mcar = 0.1, mnar = 0.1),
                            snr     = list(mar = 5, mnar = 2.5),
                            pattern = "random")
```

---

plotImps *Plot Imputed vs. Observed Values*

Description

This function will generate plots of imputed versus observed values in multiply imputed data. These plots can be examined to "sanity-check" the imputation procedure.

Usage

```
plotImps(impList,
         rMat,
         typeVec,
         targetVar   = NULL,
         interactive = FALSE)
```

Arguments

impList	A list of multiply imputed datasets.
rMat	A logical pattern matrix flagging missing values in the orginal data used to generate the imputed datasets in impList. Note that TRUE flags missing cells and FALSE flags observed cells.

typeVec	A character vector with `length(typeVec) = ncol(impList[[1]])` giving the measurement levels of the variables in `impList`. Two values are recognized: "cat" = a categorical variable (i.e., nominal or ordinal) and "con" = a continuous variable (i.e., interval or ratio).
targetVar	An optional character vector giving the column names for variables to plot. When `targetVar = NULL` all variables with imputed values are plotted.
interactive	A logical flag: Should the cycle through all plotted variables by prompting the user to continue after generating each plot?

Value

Used for its side-effects.

Author(s)

Kyle M. Lang

Examples

```
data(testImps)

plotImps(impList = testImps$impList,
         rMat    = testImps$rMat,
         typeVec = testImps$typeVec)
```

`simRegData` *Simulate Regression Data*

Description

This function will simulation regression data with known R-Squared, inter-predictor correlation, and latent grouping structure among the predictors.

Usage

```
simRegData(nObs,
           nPreds,
           r2,
           collin,
           beta,
           means           = 0.0,
           scales          = 1.0,
           itemsPerPred    = 1,
           predReliability = 0.8)
```

Arguments

nObs	An integer giving the number of rows to simulate.
nPreds	An integer giving the number of (possibly latent) predictor variables to simulate.
r2	A real number in [0, 1]. The R-Squared of the data generating model. That is, what proportion of variability in the outcome should be explained by the predictors.
collin	A real number in [-1, 1]. The correlation between the predictors (i.e., the degree of collinearity).
beta	An numeric matrix of regression coefficients with `dim(beta) = c(nPreds, 1)`. Note that the first element is taken to be the intercept.
means	A numeric vector of predictor means. Recycled, as necesssary, to match `nPreds`.
scales	A numeric vector of predictor scales. Recycled, as necesssary, to match `nPreds`.
itemsPerPred	An integer giving the number of observed items used to define each latent predictor. When `itemsPerPred = 1`, no latent structure is imposed on the predictors.
predReliability	A real number in [0, 1]. When `itemsPerPred > 1`, `predReliability` defines the proportion of reliable variance among the indicators of each latent predictor. That is, the proportion of shared variance among each latent predictor's observed indicators.

Value

An `nObs` by `nPreds * itemsPerPred + 1` data.frame of simluated data.

Note

The column labels of the simluated data will be `c("y", paste0("x", c(1 : nPreds * itemsPerPred)))`.

Author(s)

Kyle M. Lang

See Also

`imposeMissData`

Examples

```
testData <- simRegData(nObs   = 100,
                       nPreds = 10,
                       r2     = 0.5,
                       collin = 0.2,
                       beta   = matrix(c(0.25, rep(0.75, 10)))
                       )
```

Index