Package 'NestedCategBayesImpute'

April 14, 2016

Type Package

Title Synthetic Household with Structure Zeros

Version 0.1

Date 2016-02-15

Author Quanli Wang, Jerry Reiter, Jingchen Hu and Olanrewaju Akande

Maintainer Quanli Wang <quanli@stat.duke.edu>

Description
The tool set provides a set of functions to fit Dirichlet process mixture models for nested categorical data with structure zeros. It has has direct applications for generate synthetic households data.

License GPL(>=3)

LazyData TRUE

Imports Rcpp (>= 0.12.0)

LinkingTo Rcpp

R topics documented:

`checkconstraints` *Checking the possible/impossible household status under predefined constraints due to structure zeros.*

Description

Checking the possible/impossible household status under predefined constraints due to structure zeros.

Usage

```
checkconstraints(data, neededpossiblehh)
```

Arguments

`data`	A household datmatrix generated by calling samplinghouseholds.
`neededpossiblehh`	The number of possible household one need before stopping checking.

Details

Given an input household data matrix, this functions will check the possible/impossible status of each household and also output the desired number of possible/impossible households separately.

Value

A list containing information on checking result.

`outcome`	An indicator vector for possible/impossible household status under constraints.
`Households`	A data matrix for impossible households
`Index`	A vector for the original indexes of households when possible households are found. Generally not to be used.
`synHouseholds`	A data matrix for possible households.
`possible`	The actural number of possible households returned.

Author(s)

Quanli Wang

GetImpossibleHouseholds *Generate the desired number of impossible households for given household size and model parameters.*

Description

Generate the desired number of impossible households for given household size and model parameters. Also generate possible household data (synthetic data) when required.

Usage

```
GetImpossibleHouseholds(d, ACS_count, lambda, w, phi, pi, howmany, n, synindex)
```

Arguments

`d`	number of levels for each variable.
`ACS_count`	number of households for different household size. Need to change this name later.
`lambda`	lambda parameters for the household level variables.
`w`	w parameter.
`phi`	phi parameter.
`pi`	pi parameter.
`howmany`	number households to be generated at each batch. It is used as a tuning parameter for performance.
`n`	number of household in the original input data.
`synindex`	indicates if a possible household data(synthetic data) is needed.

Value

A list of variales relating to impossible household/individuals and synthetic data. Really confusing here and need more clarifiction.

`Individuals_extra`	A data matrix for individual members from the impossible households.
`HHdata_extra`	A data matrix for impossible household level variables.
`z_HHdata_individual_extra`	A data matrix for impossible household level component indicators. The household level indicators are expanded to individual members within this data matrix.
`z_HH_extra`	A vector for the household level compoment indicators.
`hh_size_new`	A vector for the number of impossible households for different household size.
`synIndividuals_all`	A data matrix for individual members from the possible(synthetic) households when synindex is non-zero. NULL otherwise.

Author(s)

Quanli Wang

groupcount *Generate 2D count table for two integer-valued vectors.*

Description

Similar to 'table' function, this function builts a contingency table of the counts at each combination of all possible values from two integer-valued input vectors.

Usage

```
groupcount(g1, g2, n1, n2)
```

Arguments

g1	The first integer-valued input vector. The max value in g1 is n1.
g2	The second integer-valued input vector. The max value in g1 is n2.
n1	The maximum value in g1.
n2	The maximum value in g2.

Details

This is implemented as an utility function to build a 2D histogram count table. For efficiency, it does not check if the maxmim values in input vectors exceed the maximum values specified.

Value

The count table.

Author(s)

Quanli Wang

Examples

```
n1 <- 20
n2 <- 10
g1 <- sample.int(n1,1000, replace = TRUE)
g2 <- sample.int(n2,1000, replace = TRUE)
counts <- groupcount(g1,g2,n1,n2)
```

`groupcount1D` *Generate histogram count for an integer-valued vector.*

Description

Generate histogram count for an integer-valued vector.

Usage

```
groupcount1D(g, n)
```

Arguments

`g`	An integer-valued input vector. The max value in g is n.
`n`	The max value in g.

Details

This is implemented as an utility function for 1D histgram count. For efficiency, it does not check if the maxmim value in input vector exceeds the maximum value specified.

Value

The count values.

Author(s)

Quanli Wang

Examples

```
n <- 20
g <- sample.int(n,1000, replace = TRUE)
counts <- groupcount1D(g,n)
```

`household` *example dataset to be used for library development.*

Description

This is the example data set that is used in Monica's paper (not exactly the same, but a new random sample of same size).

Usage

```
data("household")
```

Format

A data frame with 10000 observations on the following 9 variables.

`Hhindex` Household index

`pernum` Ask Monica to fill it in

`sex` Gender. 1 = male, 2 = female

`race` Recoded general race code. 1 = white alone, 2 = black alone,3 = American Indian/Alaska Native alone,4 = Asian or Pacific Islander alone,5 = other, 6 = two or more races.

`sthn` Gender. 1 = male, 2 = female

`age` Age. 1=0(lessthenoneyearold),2=1,...,94 = 93

`relate` 1 = head/householder, 2 = spouse, 3 = child, 4 = child-in-law, 5 = parent, 6 = parent-in-law, 7 = sibling, 8 = sibling-in-law,9 = grandchild, 10 = other relatives,11 = partner, friend, visitor,12 = other non-relatives

`ownership` Ownership of dwelling. 1 = owned or being bought (loan), 2 = rented.

`householdsize` Household size. 2=2people,3=3people,4=4people.

Details

Monica please provide the details here about the dataset.

Source

From Monica. Need real sources.

References

Jingchen Hu, Jerome P. Reiter and Quanli Wang. Dirichlet process mixture models for nested categorical data. Vol 00(000). DOI: 0000.

Examples

```
data(household)
```

---

households2individuals *Convert a household data matrix to the corresponding individual household member data matrix.*

Description

Convert a household data matrix to the corresponding individual household member data matrix.

Usage

```
households2individuals(data)
```

Arguments

`data`	household data matrix.

Value

individual household member data matrix.

Author(s)

Quanli Wang

initData *Initilize the input data structure.*

Description

Initilize the input data structure.

Usage

```
initData(household)
```

Arguments

household	A data matrix for the input household data.

Details

Curently the variables and the level of each variable are harded coded. This needs to be changed later on once it is generalized.

Value

A list object including all the necessary data variables needed by the model.

Author(s)

Quanli Wang

initOutput *Initilize output data structure for saved parameters.*

Description

Initilize output data structure for saved parameters.

Usage

```
initOutput(data, hyper, mc)
```

Arguments

data	Input data, including household info, and level of each variable.
hyper	Hyper parameters for priors.
mc	MCMC parameters.

Details

Currently, the program is harded code to the specific data format used in the the paper. Needs to be generalized once a standard foramt is defined and the general format to define structural zeros is finalized.

Value

A list of output parameters to be saved.

---

initParameters *MCMC model parameters initilization.*

Description

MCMC model parameters initilization.

Usage

```
initParameters(data, hyper)
```

Arguments

`data`	The input data to be used in the model.
`hyper`	Hyper parameters for the prior distributions.

Value

All the parameters that ae used and are to be updated by the model. Provide more details later once the individual parameters are documented.

Author(s)

Quanli Wang

---

samplehouseholds *Rcpp implementation for sampling household data without constraints.*

Description

Rcpp implementation for sampling household data without constraints.

Usage

```
samplehouseholds(phi, w, pi, d, lambda1, lambda2, currrentbatch,
                 nHouseholds, householdsize)
```

Arguments

`phi`	phi parameter
`w`	w parameter
`pi`	pi parameter
`d`	the number of levels for each variable.
`lambda1`	lambda for the first household level variable
`lambda2`	lambda for rge second household level variable
`currrentbatch`	The current batch number for the household data to be generated. The household ID will be generated based on this batch number.
`nHouseholds`	The number of households to be generated by one call to this function.
`householdsize`	The size of the household to be generated.

Details

This function allows the model to generate a batch of nHouseholds with each household of size householdsize. The generated household data will include both possible and impossible households.

Value

A data matrix with each row for one household.

Author(s)

Quanli Wang

`samplezHH` *Update household level component indexes.*

Description

Update household level component indexes.

Usage

```
samplezHH(phi, data, w, pi, S, HHdata, lambda1, lambda2)
```

Arguments

`phi`	phi parameters.
`data`	Input data.
`w`	w parameter.
`pi`	pi parameter.
`S`	household size variable.
`HHdata`	Household level variables. Currently hard-coded to two variables in the paper.
`lambda1`	lambda for household level variable 1
`lambda2`	lambda for household level variable 2 (household size)

Details

Need to add more details by Monica or Michael.

Value

A list with two variables.

`zHH` component indexes for input households

`z_HH_Individuals`
component indexes for input individuals

Author(s)

Quanli Wang

`samplezmember` *Update individual level component indezes.*

Description

Update individual level component indezes.

Usage

```
samplezmember(phi, data, w, zHH, serial)
```

Arguments

`phi`	phi parameter.
`data`	input data.
`w`	w parameters.
`zHH`	household level component indexes.
`serial`	Initial household indexes.

Value

A vector for the updated individual level component indexes.

Author(s)

Quanli Wang

`UpdateAlpha` *Update alpha parameter*

Description

Update alpha parameter.

Usage

```
UpdateAlpha(aa, ab, u)
```

Arguments

`aa`	hyper parameter a for alpha
`ab`	hyper parameter b for alpha
`u`	u parameter

Value

updated alpha parameter.

Author(s)

Quanli Wang

`UpdateBeta` *Update beta parameter*

Description

Update beta parameter.

Usage

```
UpdateBeta(ba, bb, v)
```

Arguments

`ba`	hyper parameter a for beta.
`bb`	hyper parameter b for beta.
`v`	parameter v.

Value

Updated beta parameter.

Author(s)

Quanli Wang

UpdateLambda *Update lambda parameters*

Description

Update lambda parameters.

Usage

```
UpdateLambda(dHH, K, z_HH_all, HHdata_all)
```

Arguments

dHH	A vector for number of levels for each household level variable.
K	Maximum number of components at the household level.
z_HH_all	Component indicators at the household level for all households, including all impossible households.
HHdata_all	Data matrix for household level variables for all households, including all impossible households.

Value

Updated phi parameter.

Author(s)

Quanli Wang

UpdatePhi *Update phi parameter*

Description

Update phi parameter.

Usage

```
UpdatePhi(IndividualData_all, z_Individual_all, K, L, p, d, maxd)
```

Arguments

IndividualData_all	data matrix for all individuals, including invididuals from impossible households.
z_Individual_all	component indicator matrix for all indiciduals, including individuals from impossible households.
K	Maximum number of components at the household level.
L	Maximum number of components at the individual level.

p	Number of individual level variables in the model.
d	A vector for number of levels of each individual level variable.
maxd	Maximum number of levels from all individual level variables.

Value

Updated phi parameter.

Author(s)

Quanli Wang

UpdatePi *Update pi parameter*

Description

Update pi parameter.

Usage

```
UpdatePi(alpha, z_HH_all, K)
```

Arguments

alpha	alpha parameter.
z_HH_all	component indicator at the household level for all households, including impossible households.
K	Maximum number of components at the household level.

Value

updated pi parameter.

Author(s)

Quanli wang

UpdateW *Update W and v parameters*

Description

Update W and v parameters.

Usage

```
UpdateW(beta, z_Individual_all, K, L)
```

Arguments

`beta` Beta parameter.

`z_Individual_all`
component indicator matrix for all indiciduals, including individuals from impossible households.

`K` Maximum number of components at the household level.

`L` Maximum number of components at the individual level.

Value

A list containing updated W and v parameters.

Author(s)

Quanli Wang

Index