Package 'bayesmlogit'

July 22, 2025

Type Package

Title A Multistate Life Table (MSLT) Methodology Based on Bayesian Approach

Version 1.0.1

Maintainer Xuezhixing Zhang <xuezhixing.zhang@yale.edu>

Description Create life tables with a Bayesian approach, which can be very useful for modelling a complex health process when considering multiple predisposing factors and multiple coexisting health conditions. Details for this method can be found in: Lynch, Scott, et al., (2022) <doi:10.1177/00811750221112398>; Zang, Emma, et al., (2

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports stats,ggplot2,dplyr,magrittr

Depends R (>= 4.1.0)

Suggests rmarkdown, knitr

VignetteBuilder knitr

NeedsCompilation no

Author Emma Zang [aut, cph],
Xuezhixing Zhang [aut, cre],
Scott Lynch [aut, cph]

Repository CRAN

Date/Publication 2023-11-01 17:50:20 UTC

Contents

`bayesmlogit` *Multistate Life Table Method*

Description

A Bayesian Multistate Life Table Method for survey data, developed by Lynch and Zang (2022), allowing for large state spaces with quasi-absorbing states (i.e., structural zeros in a transition matrix).

Usage

```
bayesmlogit(
  y,
  X,
  file_path = NA,
  samp = 1000,
  burn = 500,
  verbose = 100,
  thin = 5,
  trace.plot = FALSE
)
```

Arguments

`y`	A vector of state transitions, which can be created either manually or with `CreateTrans()`. See more details using `?lifedata`.
`X`	A matrix of covariates. Note that `X` must include age as a covariate.
`file_path`	The file path for outputs. If a path is specified, the result will also be saved in the given file path. You can find two result files in the specified file: `result.txt` and `resultwstep.txt`. The former contains all posterior samples generated after burn-in. The latter is sampled from the former one with a specified sampling interval.
`samp`	Number of posterior samples. For efficiency purposes, if you need a large sample (e.g., $\geq$5000), we recommend parallel computing in a cluster.
`burn`	'burn-in' period. Default is 500.
`verbose`	Progress report. Default is 10, which means this function will report the current progress for every 10 posterior samples.
`thin`	The thinning strategy to reduce autocorrelation. For example, if `thin = 5`, this function will select 1 from every 5 posterior samples and generate a new dataset named `outwstepwidth.txt`. Default is 5.
`trace.plot`	If TRUE, this function will create a new directory under given `file_path` and output corresponding trace plots using samples after burn-in.

Details

This function came from the deprecated bayeslogit package, which conducts Bayesian multinomial logistic regressions using Polya-Gamma latent variables (Polson et al. 2013). It should be jointly used with the mlifetable() function, which will generate life tables based on the estimates from regressions.

Value

A list that contains two arrays:

- **out**: An array that contains all posterior samples generated.
- **outwstepwidth**: An array generated by selecting one sample from every *thin* samples in **out**.

The number of columns in both arrays is determined by the number of covariates in X and the number of unique transition status in y. For example, if we have 12 covariates in X and 36 unique transitions in y, our result will contain (12+1)*(36-1)= 455 columns in total.

See Also

mlifeTable, lifedata, CreateTrans

Examples

```
## Not run:
data <- lifedata
y <- data[,1]
X <- data[,-1]

# This example will take about 30 mins.
out <- bayesmlogit(y, X ,samp=1000, burn=500,verbose=10)



## End(Not run)
```

CreateTrans *Create Transition Vector*

Description

A function used to create transition vectors with data in long format, which requires the dplyr package.

Usage

```
CreateTrans(ID, Age, State, Death, states)
```

Arguments

ID	A vector that specifies the ID for each subject.
Age	A vector that indicates each subject's age at this visit.
State	A vector or a factor that indicates the state for each subject at this visit.
Death	A vector that indicates whether the subject died or not at this visit.
states	The total number of states in our data.

Details

The rules for creating transitions can be found with `?lifedata`. In essence, arrange the data in long format, including details about the present state at time t. This procedure will assist in generating a dataset in long format that captures transitions by utilizing states from both time t-1 and t.

Value

A vector that contains all transitions.

See Also

lifedata

Examples

```
ID <- rep(1:50, each = 5)
Age <- rep(31:35, times = 50)
State <- sample(1:5,size=250,replace=TRUE)
Death <- rep(c(0,0,0,0,1),times=50)

Example <- data.frame(ID,Age,State,Death)

Example$trans <- CreateTrans(Example$ID,Example$Age, Example$State,Example$Death,states=6)
```

---

lifedata *Simplified Data for generating life tables.*

Description

Data extracted and processed from The Health and Retirement Study (HRS).

Usage

```
lifedata
```

Format

A data frame with 8198 rows and 16 variables:

trans Transitions that recorded in the original data. In this data, we have 6 kinds of transtions in total.

age Age for each subject.

male Sex for each subject. male=1, female=0.

black,hispanic Dummy variables for race.

mar Marital status.

educc,educg Dummy variables for education level.

cohort Birth cohort, which is birth year minus 1900.

neb,mwb,wb Dummy variables for birth regions.

nen,mwn,wn Dummy variables for residential regions.

Details

To use this package with your data, please make sure your data have a vector for transitions. The transitions can be manually created following the example below:

In `lifedata`, Each subject has 3 states in the cohort: 1: health; 2: unhealthiness; 3: death. Thus we will have 6 kind of possible transitions: 1:health to health; 2:health to unhealthiness; 3: health to death; 4: unhealthiness to health; 5: unhealthiness to unhealthiness; 6: unhealthiness to death. To check the transition for each subject, please use `lifedata[,1]`.

When creating transitions by yourself, please follow the orders as below:

	Health	Unhealthiness	Death
Health	1	2	3
Unhealthiness	4	5	6
Death	-	-	-

where the first column indicates the previous state of subjects and the first row indicates the current state that subjects are in. The numbers indicates the index of our transitions. For impossible transitions like death to death, you can also label them following the above order, which won't change the results. If transitions are not created in this order, the computation may encounter an error. One can also use `CreateTrans()` to create the transition vector.

Source

https://hrsdata.isr.umich.edu/data-products/rand?_ga=2.225225498.1006069885.1653670364-1014684070.1647264850

See Also

`CreateTrans`

`life_compare` *Compare life expectancies*

Description

A function for comparing the life expectancies of subgroups. This function will, by default, calculate the percentage of samples in your reference group with a higher (or lower) life expectancy (or proportion of total life expectancy) than other groups.

Usage

```
life_compare(
  file_path,
  file = paste(file_path, "/mplotResults", sep = ""),
  state.include = 0,
  states,
  ref.var,
  ref.level,
  index.matrix,
  prop = TRUE,
  criterion = ">",
  state.names = NA
)
```

Arguments

`file_path`	The file path for data reading. It can be inherited from `mlifetable_plot()`.
`file`	The file path for outputs. Default is `paste(file_path,"/mplotResults",sep='')`.
`state.include`	The status we aim to compare. It can be a number or a vector. Default is 0, which means we'll consider all states. It can be inherited from `mlifetable_plot()`.
`states`	The total number of states in data. It can be inherited from `mlifetable_plot()`.
`ref.var`	A vector containing all covariates used as comparison factors for each subgroup.
`ref.level`	A vector that declares the reference value of each reference variable.
`index.matrix`	A matrix that generated in `mlifeTable_plot()`. You don't need to specify it when using `mlifeTable_plot()`.
`prop`	If TRUE, this function will output the comparision reulsts of life expectancy proportions in addition to orginal comparison results. Default is TRUE. It can be inherited from `mlifetable_plot()`.
`criterion`	The criterion for comparison, which can be either ">" or "<". Default is ">".
`state.names`	A vector used to specify names of each state except death. It can be inherited from `mlifetable_plot()`.

Value

A `.csv` file with comparison results.

See Also

mlifeTable_plot

Examples

```
## Not run:

#By setting the parameter 'compare' in mlifeTable_plot() to TRUE.
#We can directly employ this function.

mlifeTable_plot(X=lifedata[,-1],state.include = 3,
      groupby = c("male","black","hispanic"),
      cred = 0.84,
      states = 3,
      file_path = ".",
      compare = TRUE,
      ref.var = c("black","hispanic"),
      ref.level = c(0,0))


## End(Not run)
```

---

mlifeTable *Multistate Life Table Method*

Description

A Bayesian Multistate Life Table Method for survey data, developed by Lynch and Zang (2022), allowing for large state spaces with quasi-absorbing states (i.e., structural zeros in a transition matrix).

Usage

```
mlifeTable(
  y,
  X,
  trans,
  states,
  file_path,
  groupby = NA,
  no_control = NA,
  values = NA,
  status = 0,
  startages = 0,
  endages = 110,
  age.gap = 1,
  nums = dim(trans)[1],
  mlifeTable_plot = FALSE,
```

```
  state.names = NA,
  ...
)
```

Arguments

y	A vector of transitions.
X	A matrix of covariates. Note that X must include age as a convariate.
trans	The posterior samples generated using ?bayesmlogit().
states	The total number of states in data.
file_path	The file path for outputs.
groupby	A vector that contains the covariates for subgroup comparisons. Default is NA, which means that we won't make subgroups.
no_control	The covariates that we don't want to control in subgroup analysis. Default is NA, which means we will control all covariates in X. As an example, in Lynch and Zang's study (2022), they incorporated education into the multinomial logit model. However, in the life table calculation, if one does not want to control for education, one could opt to use its region-specific mean rather than the sample mean using no_control.
values	A list that specifies values for covariates. Default is NA. If both no_control and values are specified, the option values takes precedence.
status	A numeric value. The option allows producing status-based life tables. Default is 0, produces population-based life tables.
startages	Start age of the life table. Default is 0.
endages	End age of the life table. Default is 110.
age.gap	This option allows users to specify the age interval of the life table. Default is 1. For example, if the survey data were sampled every 2 years, users can specify the age interval to be 2 in the life table.
nums	Number of life tables generated for each subgroup. Default is the size of posterior samples we used.
mlifeTable_plot	If TRUE, this option will create a new directory mplotResults under given file_path and output corresponding plots and tables for posterior means and credible intervals. Default is FALSE.
state.names	A vector used to specify names of each state except death. You can also specify them in the output files.
...	Extra parameters for mlifeTable_plot(). See more details using ?mlifeTable_plot().

Details

This function generates life tables based on the estimates from the Bayesian multinomial logit regressions, which can be obtained using the bayesmlogit() function. The values in the generated life table represent the expected remaining years to be spent in each state conditional on a give age. Current version was designed to only generate life tables based on data with a death state.

Value

Life tables for each subgroup.

See Also

bayesmlogit, mlifeTable_plot

Examples

```
## Not run:
#The life tables generated in the example have 3 columns, which correspond to 3 states:
#1: health; 2: unhealthiness; 3: death;

data <- lifedata
y <- data[,1]
X <- data[,-1]

# This example will take about 30 mins.

out <- bayesmlogit(y, X ,samp=1000, burn=500,verbose=10)

trans <- out$outwstepwidth
mlifeTable(y,X,trans =trans,
           groupby = c("male","black","hispanic"),
           no_control = "mar",
           startages=50,
           age.gap=1,
           states=3,
           file_path=".")

# To name each subgroup, try the subgroup.names option.
mlifeTable(y,X,trans =trans,
           groupby = c("male","black","hispanic"),
           no_control = "mar",
           states=3,
           startages=50,
           age.gap=1,
           file_path=".",
           subgroup.names= c("F-W","M-W","M-B","F-B","F-H","M-H"))

# To generate plots, try the mlifeTable_plot option
mlifeTable(y,X,trans =trans,
           groupby = c("male","black","hispanic"),
           no_control = "mar",
           states=3,
           startages=50,
           age.gap=1,
           nums = 400,
           file_path=".",
           subgroup.names= c("F-W","M-W","M-B","F-B","F-H","M-H"),
           mlifeTable_plot = T,
           cred = 0.84)
```

```
# To specify a variable at a fixed value other than the mean value. Try option "values".
mlifeTable(y,X,trans =trans,
           groupby = c("male","black","hispanic"),
           no_control = "mar",
           values = list("cohort" = 36),
           states=3,
           startages=50,
           age.gap=1,
           nums = 400,
           file_path=".",
           subgroup.names= c("F-W","M-W","M-B","F-B","F-H","M-H"),
           mlifeTable_plot = T,
           cred = 0.84)


## End(Not run)
```

`mlifeTable_plot` *Plot life expectancies*

Description

A function for plotting posterior means and their credible intervals. It can also be used as a sub-function in `mlifetable()`.

Usage

```
mlifeTable_plot(
  state.include = 0,
  groupby,
  file_path,
  X,
  cred = 0.84,
  states,
  prop = TRUE,
  subgroup.names = NULL,
  state.names = NA,
  compare = FALSE,
  midpoint.type = "mean",
  ...
)
```

Arguments

`state.include`	A vector or a number used to specify the states whose expectancy years are of interest. Default is 0, which means we'll generate plots for all states. For multiple states specified, we will get the expectancy years for each state and their sum.

groupby	A vector that contains covariates for subgroup comparisons. It can be inherited from mlifetable().
file_path	The file path for outputs. It can be inherited from mlifetable().
X	A matrix of covariates. Note that X must include age as a convariate. It can be inherited from mlifetable().
cred	Credible level. For example, if cred = 0.84, we will get the 84% credible interval.
states	The total number of states in data. It can be inherited from mlifetable().
prop	If TRUE, this function will output life expectancy proportion plots and tables in addition to original life expectancy plots. Default is TRUE.
subgroup.names	A vector that contains names of each subgroup. You can also specify them in the output files.
state.names	A vector used to specify names of each state except death. It can be inherited from mlifetable().
compare	If TRUE, this function will quote life_compare() and generate a table with all comparsion results based on the reference variables and reference levels specified. Default is FALSE.
midpoint.type	A character used to specify the midpoint type for credible interval plots. Can be either "mean" or "median". Default is "mean", which means the plots will use mean values as the middle point.
...	Extra parameters for life_compare(). See details using ?life_compare().

Value

Plots and tables for posterior means and credible intervals of each subgroups.

See Also

mlifeTable, life_compare

Examples

```
## Not run:

#Generate plots and corresponding tables only.
mlifeTable_plot(X=lifedata[,-1],state.include = 0,
      groupby = c("male","black","hispanic"),
      cred = 0.84,
      states = 3,
      file_path = ".")

#Additionally generate the comparsion results to the reference level.
mlifeTable_plot(X=lifedata[,-1],state.include = 0,
      groupby = c("male","black","hispanic"),
      cred = 0.84,
      states = 3,
      file_path = ".",
```

```
        compare = TRUE,
        ref.var = c("black","hispanic"),
        ref.level = c(0,0))

## End(Not run)
```

Index