

Package ‘lcra’

May 9, 2026

Version 1.1.5

Title Bayesian Joint Latent Class and Regression Models

Type Package

Description For fitting Bayesian joint latent class and regression models using Gibbs sampling. See the documentation for the model. The technical details of the model implemented here are described in Elliott, Michael R., Zhao, Zhangchen, Mukherjee, Bhramar, Kanaya, Alka, Needham, Belinda L., ``Methods to account for uncertainty in latent class assignments when using latent classes as predictors in regression models, with application to acculturation strategy measures" (2020) In press at Epidemiology <doi:10.1097/EDE.0000000000001139>.

License GPL-2

Encoding UTF-8

LazyData true

Biarch true

Depends R (>= 3.4.0)

Imports rlang, coda, rjags

Suggests R2WinBUGS, gtools

SystemRequirements JAGS 4.x.y or WinBUGS 1.4

URL <https://github.com/umich-biostatistics/lcra>

BugReports <https://github.com/umich-biostatistics/lcra/issues>

RoxygenNote 7.2.3

NeedsCompilation no

Author Michael Elliot [aut],
Zhangchen Zhao [aut],
Michael Kleinsasser [aut, cre]

Maintainer Michael Kleinsasser <biostat-cran-manager@umich.edu>

Repository CRAN

Date/Publication 2024-03-08 17:00:02 UTC

Contents

express	2
latent3	3
latent3_binary	4
lcra	5
paper_sim	10
paper_sim_binary	11
Index	13

express	<i>Small simulated data set</i>
---------	---------------------------------

Description

Simulated data set with continuous regression outcome. The data set contains 150 observations of 8 variables, which include 5 manifest variables, and two regressors.

Usage

```
express
```

Format

An object of class `data.frame` with 150 rows and 8 columns.

Details

- y** Discrete regression outcome of interest
- Z1** Categorical manifest variable 1
- Z2** Categorical manifest variable 2
- Z3** Categorical manifest variable 3
- Z4** Categorical manifest variable 4
- Z5** Categorical manifest variable 5
- x1** Continuous predictor variable
- x2** Continuous predictor variable

`latent3`*Simulated data set number 2 (continuous regression outcome)*

Description

Simulated data set with continuous regression outcome. The data set contains 350 observations of 16 variables, which include 12 manifest variables, and four regressors.

Usage`latent3`**Format**

An object of class `data.frame` with 350 rows and 17 columns.

Details

y Discrete regression outcome of interest

Z1 Categorical manifest variable 1

Z2 Categorical manifest variable 2

Z3 Categorical manifest variable 3

Z4 Categorical manifest variable 4

Z5 Categorical manifest variable 5

Z6 Categorical manifest variable 6

Z7 Categorical manifest variable 7

Z8 Categorical manifest variable 8

Z9 Categorical manifest variable 9

Z10 Categorical manifest variable 10

Z11 Categorical manifest variable 11

Z12 Categorical manifest variable 12

x1 Continuous predictor variable

x2 Continuous predictor variable

x3 Continuous predictor variable

x4 Continuous predictor variable

`latent3_binary`*Simulated data set number 2 (discrete regression outcome)*

Description

Simulated data set with discrete regression outcome. The data set contains 350 observations of 16 variables, which include 12 manifest variables, and four regressors.

Usage`latent3_binary`**Format**

An object of class `data.frame` with 350 rows and 17 columns.

Details

y Discrete regression outcome of interest

Z1 Categorical manifest variable 1

Z2 Categorical manifest variable 2

Z3 Categorical manifest variable 3

Z4 Categorical manifest variable 4

Z5 Categorical manifest variable 5

Z6 Categorical manifest variable 6

Z7 Categorical manifest variable 7

Z8 Categorical manifest variable 8

Z9 Categorical manifest variable 9

Z10 Categorical manifest variable 10

Z11 Categorical manifest variable 11

Z12 Categorical manifest variable 12

x1 Continuous predictor variable

x2 Continuous predictor variable

x3 Continuous predictor variable

x4 Continuous predictor variable

Description

Given a set of categorical manifest outcomes, identify unmeasured class membership among subjects, and use latent class membership to predict regression outcome jointly with a set of regressors.

Usage

```
lcra(
  formula,
  data,
  family,
  nclasses,
  manifest,
  sampler = "JAGS",
  inits = NULL,
  dir,
  n.chains = 3,
  n.iter = 2000,
  n.burnin = n.iter/2,
  n.thin = 1,
  n.adapt = 1000,
  useWINE = FALSE,
  WINE,
  debug = FALSE,
  ...
)
```

Arguments

formula	If formula = NULL, LCA without regression model is fitted. If a regression model is to be fitted, specify a formula using R standard syntax, e.g., $Y \sim \text{age} + \text{sex} + \text{trt}$. Do not include manifest variables in the regression model specification. These will be appended internally as latent classes.
data	data.frame with the column names specified in the regression formula and the manifest argument. The columns used in the regression formula can be of any type and will be dealt with using normal R behaviour. The manifest variable columns, however, must be coded as numeric using positive integers. For example, if one of the manifest outcomes takes on values 'Dislike', 'Neutral', and 'like', then code them as 1, 2, and 3.
family	a description of the error distribution to be used in the model. Currently the options are c("gaussian") with identity link and c("binomial") which uses a logit link.
nclasses	numeric, number of latent classes

<code>manifest</code>	character vector containing the names of each manifest variable, e.g., <code>manifest = c("Z1", "med_3", "X5")</code> . The values of the manifest columns must be numerically coded with levels 1 through <code>n_levels</code> , where <code>n_levels</code> is the number of levels for the <i>i</i> th manifest variable. The function will throw an error message if they are not coded properly.
<code>sampler</code>	which MCMC sampler to use? <code>lcra</code> relies on Gibbs sampling, where the options are "WinBUGS" or "JAGS". <code>sampler = "JAGS"</code> is the default, and is recommended
<code>inits</code>	list of initial values. Defaults will be set if nothing is specified. Inits must be a list with <code>n.chains</code> elements; each element of the list is itself a list of starting values for the model.
<code>dir</code>	Specify full path to the directory where you want to store the model file.
<code>n.chains</code>	number of Markov chains.
<code>n.iter</code>	number of total iterations per chain including burn-in.
<code>n.burnin</code>	length of burn-in, i.e., number of iterations to discard at the beginning. Default is <code>n.iter/2</code> .
<code>n.thin</code>	thinning rate. Must be a positive integer. Set <code>n.thin > 1</code> to save memory and computing time if <code>n.iter</code> is large.
<code>n.adapt</code>	number of adaptive samples to take when using JAGS. See the JAGS documentation for more information.
<code>useWINE</code>	logical, attempt to use the Wine emulator to run WinBUGS, defaults to FALSE on Windows and TRUE otherwise.
<code>WINE</code>	character, path to WINE binary file. If not provided, the program will attempt to find the WINE installation on your machine.
<code>debug</code>	logical, keep WinBUGS open debug, inspect chains and summary.
<code>...</code>	other arguments to <code>bugs()</code> . Run <code>?bugs</code> to see list of possible arguments to pass into <code>bugs</code> .

Details

`lcra` allows for two different Gibbs samplers to be used. The options are WinBUGS or JAGS. If you are not on a Windows system, WinBUGS can be very difficult to get working. For this reason, JAGS is the default.

For further instructions on using WinBUGS, read this:

- Microsoft Windows: no problems or additional set-up required
- Linux, Mac OS X, Unix: possible with the Wine emulator via `useWine = TRUE`. Wine is a standalone program needed to emulate a Windows system on non-Windows machines.

The manifest variable columns in **data** must be coded as numeric with positive numbers. For example, if one of the manifest outcomes takes on values 'Dislike', 'Neutral', and 'like', then code them as 1, 2, and 3.

Model Definition

The LCRA model is as follows:

$$Y_i | X_i, C_i \sim \begin{cases} N(\eta_i, \tau^{-1}) & \text{for Gaussian outcome} \\ \text{BIN}\left(1, \frac{e^{\eta_i}}{1 + e^{\eta_i}}\right) & \text{for binomial outcome} \end{cases} \text{ for } \eta_i = \beta_0 + \sum_{m=1}^P X_{im} \beta_m + \sum_{j=1}^{J-1} \alpha_j I(C_i = j)$$

$P(Z_{lj} | C_i = j) \sim \text{MULTI}(1, \pi_{1j}, \dots, \pi_{Kj})$ for the $l = 1, \dots, L$ manifest categories,
 $C_i \sim \text{MULTI}(1, \theta_1, \dots, \theta_J)$ for the latent class variable.

The following priors are the default and cannot be altered by the user:

$$P(\theta_1, \dots, \theta_J) \sim \text{DIRICHLET}(1/J, \dots, 1/J)$$

$$P(\pi_{1j}, \dots, \pi_{Kj}) \sim \text{DIRICHLET}(1/K_j, \dots, 1/K_j) \text{ for all } l = 1, \dots, L, k = 1, \dots, K$$

$$P(\alpha_j) \stackrel{\text{ind}}{\sim} N(0, 10) \text{ for all } j = 1, \dots, J$$

$$P(\beta_m) \stackrel{\text{ind}}{\sim} N(0, 10) \text{ for all } m = 1, \dots, P$$

Note that this model assumes $Y \perp Z | C$, and does not allow for dependent covariates.

Please note also that the reference category for latent classes in the outcome model output is always the J th latent class in the output, and the bugs output is defined by the Latin equivalent of the model parameters (beta, alpha, tau, pi, theta). Also, the bugs output includes the variable true, which corresponds to the MCMC draws of C_i , $i = 1, \dots, n$, as well as the MCMC draws of the deviance (DIC) statistic. Finally the bugs output for pi is stored in a three dimensional array corresponding to (class, variable, category), where category is indexed by 1 through maximum K_l ; for variables where the number of categories is less than maximum K_l , these cells will be set to NA. The parameters outputted by the lcra function currently are not user definable.

Value

Return type depends on the sampler chosen.

If sampler = "WinBUGS", then the return object is: WinBUGS object and lists of draws and summaries. Use fit\$ to browse options.

If sampler = "JAGS", then the return object is: An MCMC list of class **mcmc.list**, which can be analyzed with the coda package. Each column is a parameter and each row is a draw. You can extract a parameter by name, e.g., fit["beta[1]"]. For a list of all parameter names from the fit, call colnames(as.matrix(fit)), which returns a character vector with the names.

References

"Methods to account for uncertainty in latent class assignments when using latent classes as predictors in regression models, with application to acculturation strategy measures" (2020) In press at Epidemiology. doi:10.1097/EDE.0000000000001139

Examples

```
if(requireNamespace("rjags")){
# quick example

inits = list(list(theta = c(0.33, 0.33, 0.34), beta = rep(0, length = 3),
                alpha = rep(0, length = 2), tau = 0.5, true = rep(1, length = nrow(express))))

fit = lcra(formula = y ~ x1 + x2, family = "gaussian", data = express,
          nclasses = 3, inits = inits, manifest = paste0("Z", 1:5),
          n.chains = 1, n.iter = 50)
```

```

data('paper_sim')

# Set initial values
inits =
  list(
    list(theta = c(0.33, 0.33, 0.34), beta = rep(0, length = 3),
          alpha = rep(0, length = 2), tau = 0.5, true = rep(1, length = 100)),
    list(theta = c(0.33, 0.33, 0.34), beta = rep(0, length = 3),
          alpha = rep(0, length = 2), tau = 0.5, true = rep(1, length = 100)),
    list(theta = c(0.33, 0.33, 0.34), beta = rep(0, length = 3),
          alpha = rep(0, length = 2), tau = 0.5, true = rep(1, length = 100))
  )

# Fit model 1
fit.gaus_paper = lcra(formula = Y ~ X1 + X2, family = "gaussian",
                      data = paper_sim, nclasses = 3, manifest = paste0("Z", 1:10),
                      inits = inits, n.chains = 3, n.iter = 5000)

# Model 1 results
library(coda)

summary(fit.gaus_paper)
plot(fit.gaus_paper)

# simulated examples

library(gtools) # for Dirichlet distribution

# with binary response

n <- 500

X1 <- runif(n, 2, 8)
X2 <- rbinom(n, 1, .5)
Cstar <- rnorm(n, .25 * X1 - .75 * X2, 1)
C <- 1 * (Cstar <= .8) + 2 * ((Cstar > .8) & (Cstar <= 1.6)) + 3 * (Cstar > 1.6)

pi1 <- rdirichlet(10, c(5, 4, 3, 2, 1))
pi2 <- rdirichlet(10, c(1, 3, 5, 3, 1))
pi3 <- rdirichlet(10, c(1, 2, 3, 4, 5))

Z1<-(C==1)*t(rmultinom(n,1,pi1[1,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[1,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[1,]))%*%c(1:5)
Z2<-(C==1)*t(rmultinom(n,1,pi1[2,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[2,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[2,]))%*%c(1:5)
Z3<-(C==1)*t(rmultinom(n,1,pi1[3,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[3,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[3,]))%*%c(1:5)
Z4<-(C==1)*t(rmultinom(n,1,pi1[4,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[4,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[4,]))%*%c(1:5)
Z5<-(C==1)*t(rmultinom(n,1,pi1[5,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[5,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[5,]))%*%c(1:5)

```

```

Z6<-(C==1)*t(rmultinom(n,1,pi1[6,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[6,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[6,]))%*%c(1:5)
Z7<-(C==1)*t(rmultinom(n,1,pi1[7,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[7,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[7,]))%*%c(1:5)
Z8<-(C==1)*t(rmultinom(n,1,pi1[8,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[8,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[8,]))%*%c(1:5)
Z9<-(C==1)*t(rmultinom(n,1,pi1[9,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[9,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[9,]))%*%c(1:5)
Z10<-(C==1)*t(rmultinom(n,1,pi1[10,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[10,]))%*%c(1:5)+(C==3)*t(rmultinom(n,1,pi3[10,]))%*%c(1:5)

Z <- cbind(Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8, Z9, Z10)

Y <- rbinom(n, 1, exp(-1 - .1*X1 + X2 + 2*(C == 1) + 1*(C == 2)) /
  (1 + exp(1 - .1*X1 + X2 + 2*(C == 1) + 1*(C == 2))))

mydata = data.frame(Y, X1, X2, Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8, Z9, Z10)

inits = list(list(theta = c(0.33, 0.33, 0.34), beta = rep(0, length = 3),
  alpha = rep(0, length = 2), true = rep(1, length = nrow(mydata))))

fit = lcra(formula = Y ~ X1 + X2, family = "binomial", data = mydata,
  nclasses = 3, inits = inits, manifest = paste0("Z", 1:10),
  n.chains = 1, n.iter = 1000)

summary(fit)
plot(fit)

# with continuous response

n <- 500

X1 <- runif(n, 2, 8)
X2 <- rbinom(n, 1, .5)
Cstar <- rnorm(n, .25*X1 - .75*X2, 1)
C <- 1 * (Cstar <= .8) + 2*((Cstar > .8) & (Cstar <= 1.6)) + 3*(Cstar > 1.6)

pi1 <- rdirichlet(10, c(5, 4, 3, 2, 1))
pi2 <- rdirichlet(10, c(1, 3, 5, 3, 1))
pi3 <- rdirichlet(10, c(1, 2, 3, 4, 5))
pi4 <- rdirichlet(10, c(1, 1, 1, 1, 1))

Z1<-(C==1)*t(rmultinom(n,1,pi1[1,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[1,]))%*%c(1:5)+(C==3)*
  t(rmultinom(n,1,pi3[1,]))%*%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[1,]))%*%c(1:5)
Z2<-(C==1)*t(rmultinom(n,1,pi1[2,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[2,]))%*%c(1:5)+(C==3)*
  t(rmultinom(n,1,pi3[2,]))%*%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[2,]))%*%c(1:5)
Z3<-(C==1)*t(rmultinom(n,1,pi1[3,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[3,]))%*%c(1:5)+(C==3)*
  t(rmultinom(n,1,pi3[3,]))%*%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[3,]))%*%c(1:5)
Z4<-(C==1)*t(rmultinom(n,1,pi1[4,]))%*%c(1:5)+(C==2)*
  t(rmultinom(n,1,pi2[4,]))%*%c(1:5)+(C==3)*

```

```

t(rmultinom(n,1,pi3[4,]))**%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[4,]))**%c(1:5)
Z5<-(C==1)*t(rmultinom(n,1,pi1[5,]))**%c(1:5)+(C==2)*
t(rmultinom(n,1,pi2[5,]))**%c(1:5)+(C==3)*
t(rmultinom(n,1,pi3[5,]))**%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[5,]))**%c(1:5)
Z6<-(C==1)*t(rmultinom(n,1,pi1[6,]))**%c(1:5)+(C==2)*
t(rmultinom(n,1,pi2[6,]))**%c(1:5)+(C==3)*
t(rmultinom(n,1,pi3[6,]))**%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[6,]))**%c(1:5)
Z7<-(C==1)*t(rmultinom(n,1,pi1[7,]))**%c(1:5)+(C==2)*
t(rmultinom(n,1,pi2[7,]))**%c(1:5)+(C==3)*
t(rmultinom(n,1,pi3[7,]))**%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[7,]))**%c(1:5)
Z8<-(C==1)*t(rmultinom(n,1,pi1[8,]))**%c(1:5)+(C==2)*
t(rmultinom(n,1,pi2[8,]))**%c(1:5)+(C==3)*
t(rmultinom(n,1,pi3[8,]))**%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[8,]))**%c(1:5)
Z9<-(C==1)*t(rmultinom(n,1,pi1[9,]))**%c(1:5)+(C==2)*
t(rmultinom(n,1,pi2[9,]))**%c(1:5)+(C==3)*
t(rmultinom(n,1,pi3[9,]))**%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[9,]))**%c(1:5)
Z10<-(C==1)*t(rmultinom(n,1,pi1[10,]))**%c(1:5)+(C==2)*
t(rmultinom(n,1,pi2[10,]))**%c(1:5)+(C==3)*
t(rmultinom(n,1,pi3[10,]))**%c(1:5)+(C==4)*t(rmultinom(n,1,pi4[10,]))**%c(1:5)

Z <- cbind(Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8, Z9, Z10)

Y <- rnorm(n, 10 - .5*X1 + 2*X2 + 2*(C == 1) + 1*(C == 2), 1)

mydata = data.frame(Y, X1, X2, Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8, Z9, Z10)

inits = list(list(theta = c(0.33, 0.33, 0.34), beta = rep(0, length = 3),
alpha = rep(0, length = 2), true = rep(1, length = nrow(mydata)),
tau = 0.5))

fit = lcra(formula = Y ~ X1 + X2, family = "gaussian", data = mydata,
nclasses = 3, inits = inits, manifest = paste0("Z", 1:10),
n.chains = 1, n.iter = 1000)

summary(fit)
plot(fit)

}

```

paper_sim

Simulated data set (continuous regression outcome)

Description

Simulated data set with continuous regression outcome. The data set contains 100 observations of 13 variables, which include 10 manifest variables, and two regressors - one continuous and one dummy.

Usage

paper_sim

Format

An object of class data.frame with 100 rows and 13 columns.

Details

Y Continuous regression outcome of interest

Z1 Categorical manifest variable 1

Z2 Categorical manifest variable 2

Z3 Categorical manifest variable 3

Z4 Categorical manifest variable 4

Z5 Categorical manifest variable 5

Z6 Categorical manifest variable 6

Z7 Categorical manifest variable 7

Z8 Categorical manifest variable 8

Z9 Categorical manifest variable 9

Z10 Categorical manifest variable 10

X1 Continuous predictor variable

X2 Categorical variable with values 1, 0

paper_sim_binary *Simulated data set (discrete regression outcome)*

Description

Simulated data set with discrete regression outcome. The data set contains 100 observations of 13 variables, which include 10 manifest variables, and two regressors - one continuous and one dummy.

Usage

paper_sim_binary

Format

An object of class data.frame with 100 rows and 13 columns.

Details

- Y** Discrete regression outcome of interest
- Z1** Categorical manifest variable 1
- Z2** Categorical manifest variable 2
- Z3** Categorical manifest variable 3
- Z4** Categorical manifest variable 4
- Z5** Categorical manifest variable 5
- Z6** Categorical manifest variable 6
- Z7** Categorical manifest variable 7
- Z8** Categorical manifest variable 8
- Z9** Categorical manifest variable 9
- Z10** Categorical manifest variable 10
- X1** Continuous predictor variable
- X2** Categorical variable with values 1, 0

Index

* datasets

express, [2](#)

latent3, [3](#)

latent3_binary, [4](#)

paper_sim, [10](#)

paper_sim_binary, [11](#)

express, [2](#)

latent3, [3](#)

latent3_binary, [4](#)

lcra, [5](#)

paper_sim, [10](#)

paper_sim_binary, [11](#)