Package 'rater'

September 15, 2020

Title Fit Statistical Models of Repeated Categorical Rating Data

Version 1.0.0

Description Fit statistical models based on the Dawid-Skene model - Dawid and Skene (1979) <doi:10.2307/2346806> - to repeated categorical rating data. Full Bayesian inference for these models is supported through the Stan modelling language. 'rater' also allows the user to extract and plot key parameters of these models.

License GPL-2

URL https://jeffreypullin.github.io/rater/, https://github.com/jeffreypullin/rater

BugReports https://github.com/jeffreypullin/rater/issues

Encoding UTF-8

LazyData true

Biarch true

Depends R (>= 3.4.0)

Imports methods, Rcpp (>= 0.12.0), rstan (>= 2.19.2), rstantools (>= 2.0.0), RcppParallel (>= 5.0.1), ggplot2 (>= 2.2.1), rlang (> 0.2.0)

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), rstan (>= 2.18.1), StanHeaders (>= 2.18.0), RcppParallel (>= 5.0.1)

RoxygenNote 7.1.1

Suggests testthat, knitr, rmarkdown, covr

VignetteBuilder knitr

SystemRequirements GNU make

NeedsCompilation yes

Author Jeffrey Pullin [aut, cre] (<https://orcid.org/0000-0003-3651-5471>), Damjan Vukcevic [aut] (<https://orcid.org/0000-0001-7780-9586>)

Maintainer Jeffrey Pullin <jeffrey.pullin@gmail.com>

Repository CRAN

Date/Publication 2020-09-15 08:40:03 UTC

R topics documented:

`rater-package` *The 'rater' package.*

Description

Fit statistical models based on the Dawid-Skene model to repeated categorical rating data. Full Bayesian inference for these models is supported through the Stan modelling language. rater also allows the user to extract and plot key parameters of these models.

References

Stan Development Team (2018). RStan: the R interface to Stan. R package version 2.18.2. http://mc-stan.org

anesthesia *Anaesthetist ratings for patient suitability for surgery*

Description

The data consist of ratings, on a 4-point scale, made by four anaesthetists of patients' pre-operative health. The ratings were based on the anaesthetists assessments of a standard form completed for all of the patients. There are 45 patients (items) and four anaesthetists (raters) in total. The first anaesthetist assessed the forms a total of three times, spaced several weeks apart. The other anaesthetists each assessed the forms once. The data is in 'long' format.

Usage

```
anesthesia
```

Format

A `data.frame` with 315 rows and 3 columns:

item The item index - which item is being rated

rater The rater index - which rater is doing the rating

rating The rating given

References

Dawid, A. P., and A. M. Skene. "Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm." Applied Statistics 28, no. 1 (1979): 20.

caries *Dentist ratings of whether caries are healthy or not based on X-rays*

Description

It consists of binary ratings, made by 5 dentists, of whether a given tooth was healthy (sound) or had caries, also known as cavities. The ratings were performed using X-ray only, which was thought to be more error-prone than visual/tactile assessment of each tooth. In total 3,689 ratings were made. This data is in 'grouped' format. Each row is one of the 'pattern' with the final columns being a tally of how many times that pattern occurs in the dataset.

Usage

```
caries
```

Format

A `data.frame` with 6 columns and 32 rows.

rater_1 The rating of the dentist 1

rater_2 The rating of the dentist 2

rater_3 The rating of the dentist 3

rater_4 The rating of the dentist 4

rater_5 The rating of the dentist 5

n The number of times the rating pattern appears in the dataset

References

Espeland, Mark A., and Stanley L. Handelman. "Using Latent Class Models to Characterize and Assess Relative Error in Discrete Measurements." Biometrics 45, no. 2 (1989): 587–99.

`class_probabilities` *Extract latent class probabilities from a rater fit object*

Description

Extract latent class probabilities from a rater fit object

Usage

```
class_probabilities(fit, ...)

## S3 method for class 'mcmc_fit'
class_probabilities(fit, ...)

## S3 method for class 'optim_fit'
class_probabilities(fit, ...)
```

Arguments

`fit`	A rater fit object.
`...`	Extra arguments.

Details

The latent class probabilities are obtained by marginalising out the latent class and then calculating, for each draw of pi and theta, the conditional probability of the latent class given the other parameters and the data. Averaging these conditional probabilities gives the (unconditional) latent class probabilities retuned by this function.

Value

A I * K matrix where each element is the probably of item i being of class k. (I is the number of items and K the number of classes).

Examples

```
fit <- rater(anesthesia, "dawid_skene")
class_probabilities(fit)
```

`mcmc_diagnostics` *Retrieve MCMC convergence diagnostics for a rater fit*

Description

Retrieve MCMC convergence diagnostics for a rater fit

Usage

```
mcmc_diagnostics(fit, pars = c("pi", "theta"))
```

Arguments

`fit`	An rater `mcmc_fit` object.
`pars`	A character vector of parameter names to return. By default `c("pi","theta")`.

Details

MCMC diagnostics cannot be calculate for the z due to the marginalisation used to fit the models.

Value

A matrix where the columns represent different diagnostics and the rows are different parameters. Currently the first column contains the Rhat statistic and the second bulk effective samples size. The rownames contain the parameter names.

References

Aki Vehtari, Andrew Gelman, Daniel Simpson, Bob Carpenter, and Paul-Christian Bürkner (2019). Rank-normalization, folding, and localization: An improved R-hat for assessing convergence of MCMC. *arXiv preprint* `arXiv:1903.08008`.

See Also

`rstan::Rhat()`, `rstan::ess_bulk()`.

Examples

```
fit <- rater(anesthesia, "dawid_skene")

# Calculate the diagnostics for all parameters.
mcmc_diagnostics(fit)

# Calculate the diagnostics just for the pi parameter.
mcmc_diagnostics(fit, pars = "pi")
```

`models` *Probabilistic models of repeated categorical rating*

Description

Functions to set up models and change their prior parameters for use in `rater()`.

Usage

```
dawid_skene(alpha = NULL, beta = NULL)

hier_dawid_skene(alpha = NULL)

class_conditional_dawid_skene(alpha = NULL, beta_1 = NULL, beta_2 = NULL)
```

Arguments

`alpha`	prior parameter for pi
`beta`	prior parameter for theta
`beta_1`	First on diagonal prior probability parameter
`beta_2`	Second on diagonal prior probability parameter for theta

Value

a rater model object that can be passed to `rater()`.

plot.rater_fit *Plot a* rater_fit *object*

Description

Plot a rater_fit object

Usage

```
## S3 method for class 'rater_fit'
plot(x, pars = c("pi", "theta", "class_probabilities"), ...)
```

Arguments

x	An object of class rater_fit.
pars	A character vector of the names of the parameters to plot. By default: c("pi","theta","class_probabi
...	Other arguments. This should contain the which argument for theta plots.

Value

If one parameter is requested a ggplot2 plot. If multiple parameters are requested a list of ggplot2 plots.

Examples

```
fit <- rater(anesthesia, "dawid_skene")

# Plot all the parameters.
plot(fit)

# Select which parameters to plot.
plot(fit, pars = "pi")
```

point_estimate *Extract point estimates of parameters from a fit object*

Description

Extract point estimates of parameters from a fit object

Usage

```
point_estimate(fit, pars = c("pi", "theta", "z"), ...)
```

Arguments

`fit`	A rater fit object
`pars`	A character vector of parameter names to return. By default `c("pi","theta","z")`.
`...`	Extra arguments

Details

If the passed fit object was fit using MCMC then the posterior means are returned. If it was fit through optimisation the maximum a priori (MAP) estimates are returned. The z parameter returned is the value of class probabilities which is largest. To return the full posterior distributions of the latent class use `class_probabilities()`.

Value

A named list of the parameter estimates.

See Also

`class_probabilities()`

Examples

```
# A model fit using MCMC.
mcmc_fit <- rater(anesthesia, "dawid_skene")

# This will return the posterior mean (except for z)
post_mean_estimate <- point_estimate(mcmc_fit)

# A model fit using optimisation.
optim_fit <- rater(anesthesia, dawid_skene(), method = "optim")

# This will output MAP estimates of the parameters.
map_estimate <- point_estimate(optim_fit)
```

`posterior_interval.mcmc_fit` *Extract posterior intervals for parameters of the model*

Description

Extract posterior intervals for parameters of the model

Usage

```
## S3 method for class 'mcmc_fit'
posterior_interval(object, prob = 0.9, pars = c("pi", "theta"), ...)
```

Arguments

`object`	A rater `mcmc_fit` object.
`prob`	A single probability. The size of the credible interval returned. By default `0.9`.
`pars`	The parameters to calculate the intervals for
`...`	Other arguments.

Details

Posterior intervals can only be calculated for models fit with MCMC. In addition, posterior intervals are not meaningful for the latent class (and indeed cannot be calculated). The *full* posterior distribution of the latent class can be extracted using class_probabilities

Value

A matrix with 2 columns. The first column is the lower bound of of the credible interval and the second is the upper bound. Each row corresponds to one individuals parameters. The rownames are the parameter names.

Examples

```
fit <- rater(anesthesia, "dawid_skene", verbose = FALSE, chains = 1)

intervals <- posterior_interval(fit)
head(intervals)
```

posterior_interval.optim_fit *Extract posterior intervals for parameters of the model*

Description

Extract posterior intervals for parameters of the model

Usage

```
## S3 method for class 'optim_fit'
posterior_interval(object, prob = 0.9, pars = c("pi", "theta"), ...)
```

Arguments

object	A rater optim_fit object
prob	A probability
pars	The parameters to calculate the intervals for
...	Other arguments

posterior_samples *Extract posterior samples from a rater fit object*

Description

Extract posterior samples from a rater fit object

Usage

```
posterior_samples(fit, pars = c("pi", "theta"))
```

Arguments

fit	A rater fit object.
pars	A character vector of parameter names to return. By default `c("pi","theta")`.

Details

Posterior samples can only be returned for models fitting using MCMC not optimisation. In addition, posterior samples cannot be returned for the latent class due to the marginalisation technique used internally.

Value

A named list of the posterior samples for each parameters. For each parameter the samples are in the form returned by `rstan::extract()`.

Examples

```
fit <- rater(anesthesia, "dawid_skene")

samples <- posterior_samples(fit)

# Look at first 6 samples for each of the pi parameters
head(samples$pi)

# Look at the first 6 samples for the theta[1, 1, 1] parameter
head(samples$theta[, 1, 1, 1])

# Only get the samples for the pi parameter:
pi_samples <- posterior_samples(fit, pars = "pi")
```

---

print.mcmc_fit *Print a* mcmc_fit *object*

Description

Print a mcmc_fit object

Usage

```
## S3 method for class 'mcmc_fit'
print(x, ...)
```

Arguments

x	An object of class mcmc_fit.
...	Other arguments.

---

print.optim_fit *Print a* optim_fit *object*

Description

Print a optim_fit object

Usage

```
## S3 method for class 'optim_fit'
print(x, ...)
```

Arguments

x	An object of class optim_fit.
...	Other arguments.

---

print.rater_model *Print a* rater_model *object.*

Description

Print a rater_model object.

Usage

```
## S3 method for class 'rater_model'
print(x, ...)
```

Arguments

x	A rater_model object.
...	Other arguments

---

rater *Fit statistical models to repeated categorical rating data using Stan*

Description

This functions allows the user to fit statistical models of noisy categorical rating, based on the Dawid-Skene model, using Bayesian inference. A variety of data formats and models are supported. Inference is done using Stan, allowing models to be fit efficiently, using both optimisation and Markov Chain Monte Carlo (MCMC).

Usage

```
rater(
  data,
  model,
  method = "mcmc",
  data_format = "long",
  inits = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

data	A 2D data object: data.frame, matrix, tibble etc. with data in either long or grouped format.
model	Model to fit to data - must be rater_model or a character string - the name of the model. If the character string is used, the prior parameters will be set to their default values.
method	A length 1 character vector, either "mcmc" or "optim". This will be fitting method used by Stan. By default "mcmc"
data_format	A length 1 character vector, either "long" or "grouped". The format that the passed data is in. Defaults to "long".
inits	The initialization points of the fitting algorithm
verbose	Should rater() produce information about the progress of the chains while using the MCMC algorithm. Defaults to TRUE
...	Extra parameters which are passed to the Stan fitting interface.

Details

The default MCMC algorithm used by Stan is No U Turn Sampling (NUTS) and the default optimisation method is LGFGS. For MCMC 4 chains are run be default with 2000 iterations in total each.

Value

An object of class rater_fit containing the fitted parameters.

See Also

rstan::sampling(), rstan::optimizing()

Examples

```
# Fit a model using MCMC (the default).
mcmc_fit <- rater(anesthesia, "dawid_skene")

# Fit a model using optimisation.
optim_fit <- rater(anesthesia, dawid_skene(), method = "optim")

# Fit a model using passing data grouped data.
grouped_fit <- rater(caries, dawid_skene(), data_format = "grouped")
```

summary.mcmc_fit *Summarise a* mcmc_fit *object*

Description

Summarise a mcmc_fit object

Usage

```
## S3 method for class 'mcmc_fit'
summary(object, n_pars = 8, ...)
```

Arguments

object	An object of class mcmc_fit.
n_pars	The number of pi/theta parameters and z 'items' to display.
...	Other arguments passed to function.

summary.optim_fit *Summarise an* optim_fit *object*

Description

Summarise an optim_fit object

Usage

```
## S3 method for class 'optim_fit'
summary(object, n_pars = 8, ...)
```

Arguments

object	An object of class optim_fit.
n_pars	The number of pi/theta parameters and z 'items' to display.
...	Other arguments passed to function.

summary.rater_model *Summarise a* rater_model.

Description

Summarise a rater_model.

Usage

```
## S3 method for class 'rater_model'
summary(object, ...)
```

Arguments

object	A rater_model object.
...	Other arguments.

Index