Package ‘tidyLPA’

August 17, 2020

Type Package

Title Easily Carry Out Latent Profile Analysis (LPA) Using Open-Source or Commercial Software

Version 1.0.8

Maintainer Joshua M Rosenberg <jmrosenberg@utk.edu>

Description An interface to the 'mclust' package to easily carry out latent profile analysis ("LPA"). Provides functionality to estimate commonly-specified models. Follows a tidy approach, in that output is in the form of a data frame that can subsequently be computed on. Also has functions to interface to the commercial 'MPlus' software via the 'MplusAutomation' package.

License MIT + file LICENSE

URL https://data-edu.github.io/tidyLPA/

BugReports https://github.com/data-edu/tidyLPA/issues

Depends R (>= 2.10)

Imports dplyr, ggplot2, gtable, grid, mclust, methods, mix, MplusAutomation, tibble

Suggests knitr, lme4, missForest, parallel, pillar, rmarkdown, testthat

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Joshua M Rosenberg [aut, cre], Caspar van Lissa [aut], Jennifer A Schmidt [ctb], Patrick N Beymer [ctb], Daniel Anderson [ctb], Matthew J. Schell [ctb]

Repository CRAN

Date/Publication 2020-08-17 16:40:02 UTC
R topics documented:

- AHP
- calc_lrt
- compare_solutions
- curry_mac
- empathy
- estimate_profiles
- estimate_profiles_mclust
- estimate_profiles_mplus2
- get_data
- get_estimates
- get_fit
- id_edu
- pisaUSA15
- plot_bivariate
- plot_density
- plot_profiles
- poms
- print.tidyLPA
- print.tidyProfile
- single_imputation
- tidyLPA
- %>%

Index

AHP

Select best model using analytic hierarchy process

Description

Integrates information from several fit indices, and selects the best model.

Usage

```r
AHP(
  fitindices,
  relative_importance = c(AIC = 0.2323, AWE = 0.1129, BIC = 0.2525, CLC = 0.0922, KIC = 0.3101)
)
```

Arguments

- `fitindices`: A matrix or data.frame of fit indices, with colnames corresponding to the indices named in `relative_importance`. 
relative_importance

A named numeric vector. Names should correspond to columns in `fitindices`, and values represent the relative weight assigned to the corresponding fit index. The default value corresponds to the fit indices and weights assigned by Akogul and Erisoglu. To assign uniform weights (i.e., each index is weighted equally), assign an equal value to all.

**Details**

Many fit indices are available for model selection. Following the procedure developed by Akogul and Erisoglu (2017), this function integrates information from several fit indices, and selects the best model, using Saaty’s (1990) Analytic Hierarchy Process (AHP). Conceptually, the process consists of the following steps:

1. For each fit index, calculate the amount of support provided for each model, relative to the other models.
2. From these comparisons, obtain a "priority vector" of the amount of support for each model.
3. Compute a weighted average of the priority vectors for all fit indeces, with weights based on a simulation study examining each fit index’ ability to recover the correct number of clusters (Akogul & Erisoglu, 2016).
4. Select the model with the highest weighted average priority.

**Value**

Numeric.

**Author(s)**

Caspar J. van Lissa

**Examples**

```r
iris[,1:4] %>%
estimate_profiles(1:4) %>%
get_fit() %>%
AHP()
```

---

**calc_lrt**  
*Lo-Mendell-Rubin likelihood ratio test*

**Description**

Implements the ad-hoc adjusted likelihood ratio test (LRT) described in Formula 15 of Lo, Mendell, & Rubin (2001), or LMR LRT.

**Usage**

`calc_lrt(n, null_ll, null_param, null_classes, alt_ll, alt_param, alt_classes)`
**Arguments**

- `n` Integer. Sample size
- `null_ll` Numeric. Log-likelihood of the null model.
- `null_param` Integer. Number of parameters of the null model.
- `null_classes` Integer. Number of classes of the null model.
- `alt_ll` Numeric. Log-likelihood of the alternative model.
- `alt_param` Integer. Number of parameters of the alternative model.
- `alt_classes` Integer. Number of classes of the alternative model.

**Value**

A numeric vector containing the likelihood ratio LR, the ad-hoc corrected LMR, degrees of freedom, and the LMR p-value.

**References**


**Examples**

```r
calc_lrt(150L, -741.02, 8, 1, -488.91, 13, 2)
```

**Description**

Takes an object of class 'tidyLPA', containing multiple latent profile models with different number of classes or model specifications, and helps select the optimal number of classes and model specification.

**Usage**

```r
compare_solutions(x, statistics = "BIC")
```

**Arguments**

- `x` An object of class 'tidyLPA'.
- `statistics` Character vector. Which statistics to examine for determining the optimal model. Defaults to 'BIC'.

---

**compare_solutions**  
*Compare latent profile models*

---

**Description**

Takes an object of class 'tidyLPA', containing multiple latent profile models with different number of classes or model specifications, and helps select the optimal number of classes and model specification.

**Usage**

```r
compare_solutions(x, statistics = "BIC")
```

**Arguments**

- `x` An object of class 'tidyLPA'.
- `statistics` Character vector. Which statistics to examine for determining the optimal model. Defaults to 'BIC'.
Value

An object of class 'bestLPA' and 'list', containing a tibble of fits 'fits', a named vector 'best', indicating which model fit best according to each fit index, a numeric vector 'AHP' indicating the best model according to the AHP, an object 'plot' of class 'ggplot', and a numeric vector 'statistics' corresponding to argument of the same name.

Author(s)

Caspar J. van Lissa

Examples

iris_subset <- sample(nrow(iris), 20) # so examples execute quickly
results <- iris %>%
  subset(select = c("Sepal.Length", "Sepal.Width",
   "Petal.Length", "Petal.Width")) %>%
  estimate_profiles(1:3) %>%
  compare_solutions()

data(curry_mac)

Format

A data.frame with 1392 rows and 42 variables.

Details

sex        factor  Self-identified sex of participants, Male, Female, or Transgendered.
age_years  numeric  Participants’ age in years.
KinshipR   numeric  Mean score of moral relevance, kinship subscale.
MutualismR numeric  Mean score of moral relevance, mutualism subscale.
ExchangeR  numeric  Mean score of moral relevance, exchange subscale.
HawkR      numeric  Mean score of moral relevance, hawk subscale.
DoveR      numeric  Mean score of moral relevance, dove subscale.
DivisionR  numeric  Mean score of moral relevance, division subscale.
PossessionR numeric  Mean score of moral relevance, possession subscale.
KinshipJ   numeric  Mean score of moral judgment, kinship subscale.
References


Simulated empathy data

Description

This simulated dataset, based on Van Lissa et al., 2014, contains six annual assessments of adolescents’ mean scores on the empathic concern and perspective taking subscales of the Interpersonal Reactivity Index (Davis, 1983). The first measurement wave occurred when adolescents were, on average, 13 years old, and the last one when they were 18 years old.

Usage

data(empathy)

Format

A data frame with 467 rows and 13 variables.

Details

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Mutualism  | numeric| Mean score of moral judgment, mutualism subscale.
| Exchange   | numeric| Mean score of moral judgment, exchange subscale.
| Hawk       | numeric| Mean score of moral judgment, hawk subscale.
| Dove       | numeric| Mean score of moral judgment, dove subscale.
| Division   | numeric| Mean score of moral judgment, division subscale.
| Possession | numeric| Mean score of moral judgment, possession subscale.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>empathy</td>
<td></td>
<td>Simulated empathy data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ec1</td>
<td>numeric</td>
<td>Mean score of empathic concern in wave 1</td>
</tr>
<tr>
<td>ec2</td>
<td>numeric</td>
<td>Mean score of empathic concern in wave 2</td>
</tr>
<tr>
<td>ec3</td>
<td>numeric</td>
<td>Mean score of empathic concern in wave 3</td>
</tr>
<tr>
<td>ec4</td>
<td>numeric</td>
<td>Mean score of empathic concern in wave 4</td>
</tr>
<tr>
<td>ec5</td>
<td>numeric</td>
<td>Mean score of empathic concern in wave 5</td>
</tr>
<tr>
<td>ec6</td>
<td>numeric</td>
<td>Mean score of empathic concern in wave 6</td>
</tr>
<tr>
<td>pt1</td>
<td>numeric</td>
<td>Mean score of perspective taking in wave 1</td>
</tr>
<tr>
<td>pt2</td>
<td>numeric</td>
<td>Mean score of perspective taking in wave 2</td>
</tr>
<tr>
<td>pt3</td>
<td>numeric</td>
<td>Mean score of perspective taking in wave 3</td>
</tr>
<tr>
<td>pt4</td>
<td>numeric</td>
<td>Mean score of perspective taking in wave 4</td>
</tr>
<tr>
<td>pt5</td>
<td>numeric</td>
<td>Mean score of perspective taking in wave 5</td>
</tr>
<tr>
<td>pt6</td>
<td>numeric</td>
<td>Mean score of perspective taking in wave 6</td>
</tr>
<tr>
<td>sex</td>
<td>factor</td>
<td>Adolescent sex; M = male, F = female.</td>
</tr>
</tbody>
</table>
References


**estimate_profiles**

Estimate latent profiles

**Description**

Estimates latent profiles (finite mixture models) using the open source package mclust, or the commercial program Mplus (using the R-interface of MplusAutomation).

**Usage**

```r
estimate_profiles(
  df,
  n_profiles,
  models = NULL,
  variances = "equal",
  covariances = "zero",
  package = "mclust",
  select_vars = NULL,
  ...
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>data.frame of numeric data; continuous indicators are required for mixture modeling.</td>
</tr>
<tr>
<td>n_profiles</td>
<td>Integer vector of the number of profiles (or mixture components) to be estimated.</td>
</tr>
<tr>
<td>models</td>
<td>Integer vector. Set to NULL by default, and models are constructed from the variances and covariances arguments. See Details for the six models available in tidyLPA.</td>
</tr>
<tr>
<td>variances</td>
<td>Character vector. Specifies which variance components to estimate. Defaults to &quot;equal&quot; (constrain variances across profiles); the other option is &quot;varying&quot; (estimate variances freely across profiles). Each element of this vector refers to one of the models you wish to run.</td>
</tr>
<tr>
<td>covariances</td>
<td>Character vector. Specifies which covariance components to estimate. Defaults to &quot;zero&quot; (do not estimate covariances; this corresponds to an assumption of conditional independence of the indicators); other options are &quot;equal&quot; (estimate covariances between items, constrained across profiles), and &quot;varying&quot; (free covariances across profiles).</td>
</tr>
</tbody>
</table>
estimate_profiles

package
Character. Which package to use; 'mclust' or 'MplusAutomation' (requires Mplus to be installed). Default: 'mclust'.

select_vars
Character. Optional vector of variable names in df, to be used for model estimation. Defaults to NULL, which means all variables in df are used.

... Additional arguments are passed to the estimating function; i.e., Mclust, or mplusModeler.

Details
Six models are currently available in tidyLPA, corresponding to the most common requirements. These are:

1. Equal variances and covariances fixed to 0
2. Varying variances and covariances fixed to 0
3. Equal variances and equal covariances
4. Varying variances and equal covariances (not able to be fit w/ mclust)
5. Equal variances and varying covariances (not able to be fit w/ mclust)
6. Varying variances and varying covariances

Two interfaces are available to estimate these models; specify their numbers in the models argument (e.g., models = 1, or models = c(1,2,3)), or specify the variances/covariances to be estimated (e.g.: variances = c("equal","varying"), covariances = c("zero","equal")). Note that when mclust is used, models = c(1,2,3,6) are the only models available.

Value
A list of class 'tidyLPA'.

Examples

iris_sample <- iris[c(1:4, 51:54, 101:104), ] # to make example run more quickly

# Example 1:
iris_sample %>%
  subset(select = c("Sepal.Length", "Sepal.Width", "Petal.Length")) %>%
estimate_profiles(3)

# Example 2:
iris %>%
  subset(select = c("Sepal.Length", "Sepal.Width", "Petal.Length")) %>%
estimate_profiles(n_profiles = 1:4, models = 1:3)

# Example 3:
iris_sample %>%
  subset(select = c("Sepal.Length", "Sepal.Width", "Petal.Length")) %>%
estimate_profiles(models = c(1,2,3))
estimate_profiles_mclust

```
"Petal.Length") %>%
estimate_profiles(n_profiles = 1:4, variances = c("equal", "varying"),
covariances = c("zero", "zero"))
```

estimate_profiles_mclust

*Estimate latent profiles using mclust*

**Description**

Estimates latent profiles (finite mixture models) using the open source package *mclust*.

**Usage**

```
estimate_profiles_mclust(df, n_profiles, model_numbers, select_vars, ...)
```

**Arguments**

- **df**: data.frame with two or more columns with continuous variables
- **n_profiles**: Numeric vector. The number of profiles (or mixture components) to be estimated. Each number in the vector corresponds to an analysis with that many mixture components.
- **model_numbers**: Numeric vector. Numbers of the models to be estimated. See *estimate_profiles* for a description of the models available in tidyLPA.
- **select_vars**: Character. Optional vector of variable names in df, to be used for model estimation. Defaults to NULL, which means all variables in df are used.
- **...**: Parameters passed directly to *Mclust*. See the documentation of *Mclust*.

**Value**

An object of class ’tidyLPA’ and ’list’

**Author(s)**

Caspar J. van Lissa
estimate_profiles_mplus2

Estimate latent profiles using Mplus

Description

Estimates latent profiles (finite mixture models) using the commercial program Mplus, through the R-interface of MplusAutomation.

Usage

```r
estimate_profiles_mplus2(
  df,
  n_profiles,
  model_numbers,
  select_vars,
  ..., 
  keepfiles = FALSE
)
```

Arguments

- `df` data.frame with two or more columns with continuous variables
- `n_profiles` Numeric vector. The number of profiles (or mixture components) to be estimated. Each number in the vector corresponds to an analysis with that many mixture components.
- `model_numbers` Numeric vector. Numbers of the models to be estimated. See `estimate_profiles` for a description of the models available in tidyLPA.
- `select_vars` Character. Optional vector of variable names in `df`, to be used for model estimation. Defaults to NULL, which means all variables in `df` are used.
- `...` Parameters passed directly to `mplusModeler`. See the documentation of `mplusModeler`.
- `keepfiles` Logical. Whether to retain the files created by `mplusModeler` (e.g., for future reference, or to manually edit them).

Value

An object of class `tidyLPA` and `list`.

Author(s)

Caspar J. van Lissa
get_data

Get data objects generated by tidyLPA

Description

Get data from objects generated by tidyLPA.

Usage

get_data(x, ...)

## S3 method for class 'tidyLPA'
get_data(x, ...)

## S3 method for class 'tidyProfile'
get_data(x, ...)

Arguments

x An object generated by tidyLPA.

... further arguments to be passed to or from other methods. They are ignored in this function.

Value

If one model is fit, the data is returned in wide format as a tibble. If more than one model is fit, the data is returned in long form. See the examples.

Methods (by class)

- tidyLPA: Get data for a latent profile analysis with multiple numbers of classes and models, of class 'tidyLPA'.
- tidyProfile: Get data for a single latent profile analysis object, of class 'tidyProfile'.

Author(s)

Caspar J. van Lissa

Examples

## Not run:
if(interactive()){  
library(dplyr)  
# the data is returned in wide form  
results <- iris %>%  
  select(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) %>%  
  estimate_profiles(3)  
get_data(results)
```r
# note that if more than one model is fit, the data is returned in long form
results1 <- iris %>%
  select(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) %>%
  estimate_profiles(c(3, 4))
get_data(results1)
```

## End(Not run)

---

**get_estimates**

Get estimates from objects generated by tidyLPA

### Description

Get estimates from objects generated by tidyLPA.

### Usage

```r
get_estimates(x, ...)
```

### Arguments

- **x**
  - An object generated by tidyLPA.
- **...**
  - further arguments to be passed to or from other methods. They are ignored in this function.

### Value

A tibble.

### Methods (by class)

- **tidyLPA**: Get estimates for a latent profile analysis with multiple numbers of classes and models, of class 'tidyLPA'.
- **tidyProfile**: Get estimates for a single latent profile analysis object, of class 'tidyProfile'.

### Author(s)

Caspar J. van Lissa
get_fit

### Examples

```r
## Not run:
if(interactive()){
  results <- iris %>%
    select(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) %>%
    estimate_profiles(3)
  get_estimates(results)
  get_estimates(results[[1]])
}

## End(Not run)
```

---

**get_fit**

*Get fit indices from objects generated by tidyLPA*

**Description**

Get fit indices from objects generated by tidyLPA.

**Usage**

```r
get_fit(x, ...)  
## S3 method for class 'tidyLPA'
get_fit(x, ...)
## S3 method for class 'tidyProfile'
get_fit(x, ...)
```

**Arguments**

- **x**
  - An object generated by tidyLPA.
- **...**
  - Further arguments to be passed to or from other methods. They are ignored in this function.

**Value**

A tibble. Learn more at https://data-edu.github.io/tidyLPA/articles/Introduction_to_tidyLPA.html#getting-fit-statistics

**Methods (by class)**

- `tidyLPA`: Get fit indices for a latent profile analysis with multiple numbers of classes and models, of class `tidyLPA`.
- `tidyProfile`: Get fit indices for a single latent profile analysis object, of class `tidyProfile`.
Author(s)
Caspar J. van Lissa

Examples

```r
## Not run:
if(interactive()){
  results <- iris %>%
    select(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) %>%
    estimate_profiles(3)
  get_fit(results)
  get_fit(results[[1]])
}
## End(Not run)
```

---

**id_edu**

*Simulated identity data*

Description

This simulated dataset, based on Crochetti et al., 2014, contains five annual assessments of adolescents’ mean scores on the commitment, exploration (in depth), and reconsideration subscales of the Utrecht-Management of Identity Commitments Scale (Crocetti et al., 2008). The scores reported here reflect the educational identity subscales of this instrument. The first measurement wave occurred when adolescents were, on average, 14 years old, and the last one when they were 18 years old.

Usage

```r
data(id_edu)
```

Format

A data frame with 443 rows and 16 variables.

Details

<table>
<thead>
<tr>
<th>Column</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>com1</td>
<td>numeric</td>
<td>Mean score of educational commitment in wave 1</td>
</tr>
<tr>
<td>exp1</td>
<td>numeric</td>
<td>Mean score of educational exploration in wave 1</td>
</tr>
<tr>
<td>rec1</td>
<td>numeric</td>
<td>Mean score of educational reconsideration in wave 1</td>
</tr>
<tr>
<td>com2</td>
<td>numeric</td>
<td>Mean score of educational commitment in wave 2</td>
</tr>
<tr>
<td>exp2</td>
<td>numeric</td>
<td>Mean score of educational exploration in wave 2</td>
</tr>
<tr>
<td>rec2</td>
<td>numeric</td>
<td>Mean score of educational reconsideration in wave 2</td>
</tr>
<tr>
<td>com3</td>
<td>numeric</td>
<td>Mean score of educational commitment in wave 3</td>
</tr>
<tr>
<td>exp3</td>
<td>numeric</td>
<td>Mean score of educational exploration in wave 3</td>
</tr>
<tr>
<td>rec3</td>
<td>numeric</td>
<td>Mean score of educational reconsideration in wave 3</td>
</tr>
</tbody>
</table>
References


pisaUSA15

student questionnaire data with four variables from the 2015 PISA for students in the United States

Description

student questionnaire data with four variables from the 2015 PISA for students in the United States

Usage

pisaUSA15

Format

Data frame with columns #'

broad_interest  composite measure of students’ self reported broad interest
enjoyment  composite measure of students’ self reported enjoyment
instrumental_mot  composite measure of students’ self reported instrumental motivation
self_efficacy  composite measure of students’ self reported self efficacy ...

Source

http://www.oecd.org/pisa/data/
plot_bivariate  
Create correlation plots for a mixture model

Description

Creates a faceted plot of two-dimensional correlation plots and unidimensional density plots for an object of class 'tidyProfile'.

Usage

plot_bivariate(
  x,
  variables = NULL,
  sd = TRUE,
  cors = TRUE,
  rawdata = TRUE,
  bw = FALSE,
  alpha_range = c(0, 0.1),
  return_list = FALSE
)

Arguments

x  
tidyProfile object to plot. A tidyProfile is one element of a tidyLPA analysis.

variables  
Which variables to plot. If NULL, plots all variables that are present in all models.

sd  
Logical. Whether to show the estimated standard deviations as lines emanating from the cluster centroid.

cors  
Logical. Whether to show the estimated correlation (standardized covariance) as ellipses surrounding the cluster centroid.

rawdata  
Logical. Whether to plot raw data, weighted by posterior class probability.

bw  
Logical. Whether to make a black and white plot (for print) or a color plot. Defaults to FALSE, because these density plots are hard to read in black and white.

alpha_range  
Numeric vector (0-1). Sets the transparency of geom_density and geom_point.

return_list  
Logical. Whether to return a list of ggplot objects, or just the final plot. Defaults to FALSE.

Value

An object of class 'ggplot'.

Author(s)

Caspar J. van Lissa
plot_density

Examples

# Example 1
iris_sample <- iris[c(1:10, 51:60, 101:110), ] # to make example run more quickly
## Not run:
iris_sample %>%
  subset(select = c("Sepal.Length", "Sepal.Width")) %>%
  estimate_profiles(n_profiles = 2, models = 1) %>%
  plot_bivariate()

## End(Not run)
# Example 2
## Not run:
mtcars %>%
  subset(select = c("wt", "qsec", "drat")) %>%
  poms() %>%
  estimate_profiles(3) %>%
  plot_bivariate()

## End(Not run)

plot_density

Create density plots for mixture models

Description

Creates a faceted plot of density plots for an object of class 'tidyLPA'. For each variable, a Total density plot will be shown, along with separate density plots for each latent class, where cases are weighted by the posterior probability of being assigned to that class.

Usage

plot_density(
  x,
  variables = NULL,
  bw = FALSE,
  conditional = FALSE,
  alpha = 0.2,
  facet_labels = NULL
)

Arguments

  x  Object to plot.
  variables Which variables to plot. If NULL, plots all variables that are present in all models.
  bw Logical. Whether to make a black and white plot (for print) or a color plot. Defaults to FALSE, because these density plots are hard to read in black and white.
conditional Logical. Whether to show a conditional density plot (surface area is divided amongst the latent classes), or a classic density plot (surface area of the total density plot is equal to one, and is subdivided amongst the classes).

alpha Numeric (0-1). Only used when bw and conditional are FALSE. Sets the transparency of geom_density, so that classes with a small number of cases remain visible.

facet_labels Named character vector, the names of which should correspond to the facet labels one wishes to rename, and the values of which provide new names for these facets. For example, to rename variables, in the example with the 'iris' data below, one could specify: facet_labels = c("Pet_leng" = "Petal length").

Value
An object of class 'ggplot'.

Author(s)
Caspar J. van Lissa

Examples
## Not run:
results <- iris %>%
  subset(select = c("Sepal.Length", "Sepal.Width",
                  "Petal.Length", "Petal.Width")) %>%
  estimate_profiles(1:3)
## End(Not run)
## Not run:
plot_density(results, variables = "Petal.Length")
## End(Not run)
## Not run:
plot_density(results, bw = TRUE)
## End(Not run)
## Not run:
plot_density(results, bw = FALSE, conditional = TRUE)
## End(Not run)
## Not run:
plot_density(results[[2]], variables = "Petal.Length")
## End(Not run)
Description

Creates a profile plot according to best practices, focusing on the visualization of classification uncertainty by showing:

1. Bars reflecting a confidence interval for the class centroids
2. Boxes reflecting the standard deviations within each class; a box encompasses +/- 64% of the observations in a normal distribution
3. Raw data, whose transparency is weighted by the posterior class probability, such that each datapoint is most clearly visible for the class it is most likely to be a member of.

Usage

```r
plot_profiles(
  x,
  variables = NULL,
  ci = 0.95,
  sd = TRUE,
  add_line = TRUE,
  rawdata = TRUE,
  bw = FALSE,
  alpha_range = c(0, 0.1),
  ...
)
```

## Default S3 method:
```r
plot_profiles(
  x,
  variables = NULL,
  ci = 0.95,
  sd = TRUE,
  add_line = FALSE,
  rawdata = TRUE,
  bw = FALSE,
  alpha_range = c(0, 0.1),
  ...
)
```

Arguments

- `x` An object containing the results of a mixture model analysis.
- `variables` A character vectors with the names of the variables to be plotted (optional).
ci Numeric. What confidence interval should the errorbars span? Defaults to a 95% confidence interval. Set to NULL to remove errorbars.

sd Logical. Whether to display a box encompassing +/- 1SD Defaults to TRUE.

add_line Logical. Whether to display a line, connecting cluster centroids belonging to the same latent class. Defaults to TRUE. Note that the additional information conveyed by such a line is limited.

rawdata Should raw data be plotted in the background? Setting this to TRUE might result in long plotting times.

bw Logical. Should the plot be black and white (for print), or color?

alpha_range The minimum and maximum values of alpha (transparancy) for the raw data. Minimum should be 0; lower maximum values of alpha can help reduce overplotting.

... Arguments passed to and from other functions.

Value

An object of class 'ggplot'.

Author(s)

Caspar J. van Lissa

Examples

# Example 1
iris_sample <- iris[c(1:10, 51:60, 101:110), ] # to make example run more quickly
iris_sample %>%
  subset(select = c("Sepal.Length", "Sepal.Width")) %>%
  estimate_profiles(n_profiles = 1:2, models = 1:2) %>%
  plot_profiles()

# Example 2

mtcars %>%
  subset(select = c("wt", "qsec", "drat")) %>%
poms() %>%
estimate_profiles(1:4) %>%
plot_profiles(add_line = F)
poms

Apply POMS-coding to data

Description
Takes in a data.frame, and applies POMS (proportion of maximum)-coding to the numeric columns.

Usage
poms(data)

Arguments
data A data.frame.

Value
A data.frame.

Author(s)
Caspar J. van Lissa

Examples
data <- data.frame(a = c(1, 2, 2, 4, 1, 6),
                   b = c(6, 6, 3, 5, 3, 4),
                   c = c("a", "b", "b", "t", "f", "g"))
poms(data)

print.tidyLPA

Print tidyLPA

Description
S3 method 'print' for class 'tidyLPA'.

Usage
## S3 method for class 'tidyLPA'
print(
  x,
  stats = c("AIC", "BIC", "Entropy", "prob_min", "prob_max", "n_min", "n_max", "BLRT_p"),
  digits = 2,
  na.print = "",
  ...
)
Arguments

x An object of class 'tidyLPA'.
digits minimal number of significant digits, see print.default.
na.print a character string which is used to indicate NA values in printed output, or NULL. See print.default.
... further arguments to be passed to or from other methods. They are ignored in this function.

Author(s)

Caspar J. van Lissa

Examples

## Not run:
if(interactive()){
  iris %>%
    select(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) %>%
    estimate_profiles(3)
}
## End(Not run)

print.tidyProfile  Print tidyProfile

Description

S3 method 'print' for class 'tidyProfile'.

Usage

## S3 method for class 'tidyProfile'
print(x, digits = 2, na.print = "", ...)

Arguments

x An object of class 'tidyProfile'.
digits minimal number of significant digits, see print.default.
na.print a character string which is used to indicate NA values in printed output, or NULL. See print.default.
... further arguments to be passed to or from other methods. They are ignored in this function.
single_imputation

Author(s)
Caspar J. van Lissa

Examples

```r
## Not run:
if(interactive()){
  tmp <- iris %>%
    select(Sepal.Length, Sepal.Width, Petal.Length, Petal.Width) %>%
    estimate_profiles(3)
  tmp[[2]]
}
## End(Not run)
```

Description

This function accommodates several methods for single imputation of data. Currently, the following methods are defined:

- "imputeData" Applies the mclust native imputation function `imputeData`
- "missForest" Applies non-parametric, random-forest based data imputation using `missForest`. Random forests can accommodate any complex interactions and non-linear relations in the data. My simulation studies indicate that this method is preferable to mclust’s `imputeData` (see examples).

Usage

```r
single_imputation(x, method = "imputeData")
```

Arguments

- `x` A data.frame or matrix.
- `method` Character. Imputation method to apply, Default: 'imputeData'

Value

A data.frame

Author(s)
Caspar J. van Lissa
Examples

```r
## Not run:
library(ggplot2)
library(missForest)
library(mclust)

dm <- 2
k <- 3
n <- 100
V <- 4

# Example of one simulation
class <- sample.int(k, n, replace = TRUE)
dat <- matrix(rnorm(n*V, mean = (rep(class, each = V)-1)*dm), nrow = n,
               ncol = V, byrow = TRUE)
results <- estimate_profiles(data.frame(dat), 1:5)
plot_profiles(results)
compare_solutions(results)

# Simulation for parametric data (i.e., all assumptions of latent profile
# analysis met)
simulation <- replicate(100, {
  class <- sample.int(k, n, replace = TRUE)
dat <- matrix(rnorm(n*V, mean = (rep(class, each = V)-1)*dm), nrow = n,
               ncol = V, byrow = TRUE)
d <- prodNA(dat)
d_mf <- missForest(d)$ximp
m_mf <- Mclust(d_mf, G = 3, "EEI")
d_im <- imputeData(d, verbose = FALSE)
m_im <- Mclust(d_im, G = 3, "EEI")

class_tabl_mf <- sort(prop.table(table(class, m_mf$classification)),
                       decreasing = TRUE)[1:3]
class_tabl_im <- sort(prop.table(table(class, m_im$classification)),
                       decreasing = TRUE)[1:3]
c(sum(class_tabl_mf), sum(class_tabl_im))
})
rowMeans(simulation)
# Performance SD
colSD(t(simulation))
# Plot shows slight advantage for missForest
plotdat <- data.frame(accuracy = as.vector(simulation), model =
                      rep(c("mf", "im"), n))
ggplot(plotdat, aes(x = accuracy, colour = model))+geom_density()

# Simulation for real data (i.e., unknown whether assumptions are met)
simulation <- replicate(100, {
  d <- prodNA(iris[,1:4])
})
```

single_imputation
tidyLPA <- missForest(d$ximp)
m_mf <- Mclust(d_mf, G = 3, "EEI")
d_im <- imputeData(d, verbose = FALSE)
m_im <- Mclust(d_im, G = 3, "EEI")

class_tabl_mf <- sort(prop.table(table(iris$Species, 
m_mf$classification)), decreasing = TRUE)[1:3]
class_tabl_im <- sort(prop.table(table(iris$Species, 
m_im$classification)), decreasing = TRUE)[1:3]
c(sum(class_tabl_mf), sum(class_tabl_im))

# Performance on average
rowMeans(simulation)
# Performance SD
colSD(t(simulation))
# Plot shows slight advantage for missForest
plotdat <- data.frame(accuracy = as.vector(tmp),
                       model = rep(c("mf", "im"), n))
ggplot(plotdat, aes(x = accuracy, colour = model))+geom_density()

## End(Not run)

tidyLPA  

**Description**

Latent Profile Analysis (LPA) is a statistical modeling approach for estimating distinct profiles, or groups, of variables. In the social sciences and in educational research, these profiles could represent, for example, how different youth experience dimensions of being engaged (i.e., cognitively, behaviorally, and affectively) at the same time.

**Details**

tidyLPA provides the functionality to carry out LPA in R. In particular, tidyLPA provides functionality to specify different models that determine whether and how different parameters (i.e., means, variances, and covariances) are estimated and to specify (and compare solutions for) the number of profiles to estimate.

```r
%>%
Pipe
```

**Description**

tidyLPA suggests using the pipe operator, %>%, from the magrittr package (imported here from the dplyr package).
Arguments

lhs, rhs
An object and a function to apply to it

Examples

# Instead of
# you can write
iris %>%
Index

* correlation
  plot_bivariate, 16
* datasets
  curry_mac, 5
  empathy, 6
  id_edu, 14
  pisaUSA15, 15
* density
  plot_density, 17
* mixture
  plot_bivariate, 16
  plot_density, 17
  plot_profiles, 19
* plot
  plot_bivariate, 16
  plot_density, 17
  plot_profiles, 19
%>%, 25

AHP, 2, 5

calc_lrt, 3
compare_solutions, 4
curry_mac, 5

empathy, 6
estimate_profiles, 7, 9, 10
estimate_profiles_mclust, 9
estimate_profiles_mplus2, 10

get_data, 11
get_estimates, 12
get_fit, 13

id_edu, 14
imputeData, 23

Mclust, 8, 9
mclust, 7, 9
missForest, 23
MplusAutomation, 7, 10

mplusModeler, 8, 10
pisaUSA15, 15
plot_bivariate, 16
plot_density, 17
plot_profiles, 19
poms, 21
print.default, 22
print.tidyLPA, 21
print.tidyProfile, 22

single_imputation, 23
tidyLPA, 25