Package ‘cacIRT’

October 12, 2022

Type Package

Title Classification Accuracy and Consistency under Item Response Theory

Version 1.4

Date 2015-08-15

Author Quinn N. Lathrop

Maintainer Quinn N. Lathrop <quinn.lathrop@gmail.com>

Description Computes classification accuracy and consistency indices under Item Response Theory. Implements the total score IRT-based methods in Lee, Hanson & Brennan (2002) and Lee (2010), the IRT-based methods in Rudner (2001, 2005), and the total score nonparametric methods in Lathrop & Cheng (2014). For dichotomous and polytomous tests.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2015-08-28 01:08:33

R topics documented:

  cacIRT-package ......................................................... 2
  class.Lee ................................................................. 3
  class.Rud ................................................................. 5
  Lee.poly ................................................................. 6
  Nonparametric Approach to CA and CC ............................... 8
  recursive.raw .......................................................... 10
  Useful IRT Functions ................................................ 12

Index 14
Description

Computes classification accuracy and consistency under Item Response Theory by the approach proposed by Lee, Hanson & Brennan (2002) and Lee (2010), the approach proposed by Rudner (2001, 2005), and the approach proposed by Lathrop & Cheng (2014).

Details

- **Package**: cacIRT
- **Type**: Package
- **Version**: 1.3
- **Date**: 2015-08-15
- **License**: GPL (>= 2)

This package computes classification accuracy and consistency indices with two approaches proposed by Lee, Hanson & Brennan (2002) and Lee (2010) or by Rudner (2001, 2005). The two functions `class.Lee()` and `class.Rud()` are the wrapper functions for the most common implementations of the respective approaches. They accept a range of inputs: ability estimates, quadrature points, or response data matrix and item parameters. Marginal indices are computed with either the D (using a theoretical or simulated distribution) or P (using the sample directly) method (see Lee (2010)). The function `recursive.raw()` computes the probabilities of total scores given ability and item parameters and may be of interest outside of classification.

The major difference between the Lee approach and the Rudner approach is the scale that the classification occurs on. The Lee approach uses the total score scale, and finds the probability of each total score given an examinee’s latent ability estimate and the item parameters. The cut score is also given as a total score. The Rudner approach occurs on the latent trait scale, and is given a cut score on the latent trait scale. Despite their similarities, the two estimators generally do not estimate the same index, see Lathrop & Cheng (2013) and Lathrop (2015) for discussion and simulation studies.

A new nonparametric approach is also provided with `pnr()` and `Lee.pnr()`. It is a nonparametric extension to the Lee approach and is explained and tested in Lathrop & Cheng (2014). This approach does not require an assumption of a parametric IRT model or a parametric ability distribution and is often more accurate when those assumptions are violated compared to parametric approaches.

Polytomous tests (where item responses are in more categories than two ordered categories) are easily computed with `Lee.pnr()` and `class.Rud`. To use Lee’s (2010) approach with polytomous or mixed format tests, use `Lee.poly.P()`, `Lee.poly.D()`, and/or `gen.rec.raw()`.

Author(s)

Quinn N. Lathrop
class.Lee

Maintainer: <quinn.lathrop @ gmail.edu>

References


class.Lee          Computes classification accuracy and consistency with Lee’s approach.

Description

Computes classification accuracy and consistency with Lee’s approach. The probability of each possible total score conditional on ability is found with recursive.raw. Those probabilities are grouped according to the cut scores and used to estimate the indices. See references or code for details.

Usage

class.Lee(cutscore, ip, ability = NULL, rdm = NULL, quadrature = NULL, D = 1.7)
Lee.D(cutscore, ip, quadrature, D = 1.7)
Lee.P(cutscore, ip, theta, D = 1.7)

Arguments

cutscore       A scalar or vector of cut scores on the True Score scale. If you have cut scores on the theta scale, you can transform them with irf (See example for irf). Should not include 0 or the max total score, as the end points are added internally.

ip             Matrix of item parameters, columns are discrimination, difficulty, guessing, respectively. For 1PL and 2PL, still give a Jx3 matrix, with ip[,1] = 1 and ip[,3] = 0 for the 1PL for example.

ability, theta  Ability estimates for each subject.
**class.Lee**

R function to calculate classification consistency and accuracy for complex assessments using item response theory.

- **rdm** The response data matrix with rows as subjects and columns as items
- **quadrature** A list containing 1) The quadrature points and 2) Their corresponding weights
- **D** Scaling constant for IRT parameters, defaults to 1.7, alternatively often set to 1.

**Details**

Must give only one ability, rdm, or quadrature. If ability is given, those scores are used for the P method. If rdm is given, ability is estimated with MLE (perfect response patterns given a -4 or 4) and used for the P method. If quadrature, the D method is used. class.Lee calls Lee.D or Lee.P.

**Value**

- **Marginal** A matrix with two columns of marginal accuracy and consistency per cut score (and simultaneous if multiple cutscores are given)
- **Conditional** A list of two matrixes, one for conditional accuracy and one for conditional consistency. Each matrix has one row per subject (or quadrature point).

**Note**

In order to score above a cut, an examinee must score at or above the cut score. Since we are working on the total score scale, be aware that if a cut score is given with a decimal (like 2.4), the examinee must have a total score at the next integer or more (so 3 or more) to score above the cut.

**Author(s)**

Quinn N. Lathrop

**References**


**Examples**

```r
# from rdm, item parameters denote 4 item 1PL test, cut score at x=2
# only print marginal indices
params <- matrix(c(1,1,1,-2,1,0,1,0,0,0,0,0,4,3)
rdm <- sim(params, rnorm(100))
class.Lee(2, params, rdm = rdm)$Marginal

# or from 40 quadrature points and weights, 2 cut scores
quad <- normal.qu(40)
class.Lee(c(2,3), params, quadrature = quad, D = 1)$Marginal
```
class.Rud

Description

Computes classification accuracy and consistency with Rudner's approach. For each examinee, a normal distribution is created with mean at the ability estimate and standard deviation equal to the standard error of the ability estimate. Rudner's method assumes the standard error is conditionally normally distributed. The area under this normal curve between cut scores is used to estimate the indices. See references.

Usage

class.Rud(cutscore, ip, ability = NULL, se = NULL, rdm = NULL, quadrature = NULL, D = 1.7)
Rud.D(cutscore, quadrature, sem)
Rud.P(cutscore, theta, sem)

Arguments

cutscore A scalar or vector of cut scores on the theta scale. Should not include +- inf, the function will include them.
ip Matrix of item parameters, columns are discrimination, difficulty, guessing. For 1PL and 2PL, still give a Jx3 matrix, with ip[,1] = 1 and ip[,3] = 0 for example.
ability, theta Ability estimates for each subject.
se, sem Standard errors of ability estimates
rdm The response data matrix with rows as subjects and columns as items
quadrature A list containing [[1]] The quadrature points and [[2]] Their corresponding weights
D The scaling constant for the IRT parameters, defaults to 1.7, alternatively often set to 1.

Details

Must give only ability and se, rdm, or quadrature. If ability and se are given, those scores are used for the P method. If rdm is given, ability and se are estimated with MLE (perfect response patterns given a -4 or 4) and used for the P method. If quadrature, the D method is used.

Value

Marginal A matrix with two columns of marginal accuracy and consistency per cut score and/or simultaneous
Conditional A list of two matrixes, one for conditional accuracy and one for conditional consistency. Each matrix has one row per subject (or quadrature point).
Note

class.Rud is a wrapper for Rud.P and Rud.D.

Author(s)

Quinn Lathrop

References


Examples

```r
##from rdm, item parameters denote 4 item 1PL test, cut score at theta=.5
##only return marginal indices

params<-matrix(c(1,1,1,1,-2,1,0,1,0,0,0,0),4,3)
rdm<-sim(params, rnorm(100))

class.Rud(.5, params, rdm = rdm)$Marginal

##or from 40 quadrature points and weights, 2 cut scores

quad <- normal.qu(40)

class.Rud(c(-.5,1.5), params, quadrature = quad, D = 1)$Marginal
```

---

**Lee.poly**

*Computes classification accuracy and consistency with Lee’s approach for polytomous IRT models.*

Description

Computes classification accuracy and consistency with Lee’s approach for polytomous tests. The probability of each possible total score conditional on ability is found with gen.rec.raw(). Those probabilities are grouped according to the cut scores and used to estimate the indices.

Usage

```
Lee.poly.D(cutscore, Pij, quadrature)
Lee.poly.P(cutscore, Pij, theta)
```
Arguments

cutscore  A scalar or vector of cut scores on the True Score scale. If you have cut scores on the theta scale, you can transform them with irf (See example for irf). Should not include 0 or the max total score, as the end points are added internally.

Pij  An NxMxJ array of probabilities. Each slice of the array represents an item. Within a slice, each row corresponds to the respective element in theta and each column represents a response category from 0, 1, ..., M. At a minimum, M=1, in which case the array is Nx2xJ and represents the dichotomous item case.

theta  Ability estimates for each subject. Must correspond to the first dimension of Pij.

quadrature  A list containing 1) The quadrature points and 2) Their corresponding weights. Must correspond to the first dimension of Pij.

Details

The polytomous generalization to class.Lee. Requires the user build the Pij array.

Value

Marginal  A matrix with two columns of marginal accuracy and consistency per cut score (and simultaneous if multiple cutscores are given)

Conditional  A matrix of conditional accuracy and conditional consistency

Note

In order to score above a cut, an examinee must score at or above the cut score. Since we are working on the total score scale, be aware that if a cut score is given with a decimal (like 2.4), the examinee must have a total score at the next integer or more (so 3 or more) to score above the cut.

If the test is mixed format (some dichotomous, some polytomous items), Pij must be of an appropriate size for the item with the most response categories. The response categories that do no appear in other items can be filled with zeros. Note also that the function assumes response categories are scored as 0,1,2,3,...,M

Note

While this function is needed for polytomous tests for the Lee approach, class.Rud() works directly with polytomous tests when given the ability estimate and the standard error and so does not need an analogous set of functions.

Author(s)

Quinn N. Lathrop

References

Examples

# Same example as \texttt{class.Lee()},
# build \texttt{Pij} the same as in the example for \texttt{gen.rec.raw()}.  

params <- matrix(c(1,1,1,-2,1,0,0,0,0,0,0,4,3)
theta <- rnorm(100)

Pij.flat <- irf(params, theta)$f
Pij.array <- array(NA, dim = c(length(theta), 2, nrow(params)))
Pij.array[,1,] <- 1 - Pij.flat  # P(X_j = 0 | theta_i)
Pij.array[,2,] <- Pij.flat     # P(X_j = 1 | theta_i)

Lee.poly.P(2, Pij.array, theta)$Marginal

# in the dichotomous case, identical to \texttt{Lee.P()}
Lee.P(2, params, theta)$Marginal

# For Rudner and polytomous tests, compute the theta estimate and se and use those as input
theta.est <- theta
# just for example
theta.se <- SEM(params, theta.est)
# also for example, \texttt{SEM()} assumes 3PL model,
# but if you use mirt or similar package,
# the theta estimates and their se will be available
Rud.P(.5, theta.est, theta.se)$Marginal

---

Nonparametric Approach to CA and CC  

Description  
Computes classification accuracy and consistency with Lathrop & Cheng’s (2014) approach. First, the kernel-smoothed estimate of the probability of a correct response, conditional on observed total score, is found with \texttt{pnr()}. Then, the method proceeds similar to \texttt{class.Lee()}. Using the nonparametric approach does not require a parametric IRT model, keeps the problem on the total score scale, and can produce more accurate CA and CC estimates when the IRT model’s assumptions are violated (see Lathrop & Cheng, 2014).

Usage  
\texttt{Lee.pnr(cutscore, pnr.out)}  
\texttt{pnr(resp, bw.g = NULL, alpha = .5)}
Nonparametric Approach to CA and CC

Arguments

cutscore
A scalar or vector of cut scores on the total score scale. Should not include 0 or
the max total score, as the end points are added internally.

pnr.out
The output from pnr(). It is a list of length 3 where
pnr.out[[1]] is a vector of T evaluation points on the total score scale (integers
from 0 to the max total score)
pnr.out[[2]] is a vector of the observed density at each evaluation point
pnr.out[[3]] is a TxMxJ array. Each slice is an item. Within a slice, rows are
for evaluation points and columns are for the probability of the score category.
This has a similar structure to Pij seen in Lee.poly()

resp
The response data matrix with rows as subjects and columns as items. Because
the method is based on total score, the method is not robust to missing data. Any
NA in resp will propagate to the output.

bw.g
The global bandwidth parameter. The default of NULL will estimate the global
bandwidth with the optimal (in terms of MSE) estimate of the bandwidth for
normally distributed variables. The default is generally a good starting point.

alpha
The adaptivity of the bandwidth parameter. A value of 0 means no adaptation
and each evaluation point uses the value in bw.g. For, other values (up to and
including 1), the bandwidth parameter will shrink if the evaluation point is in
an area of high density and grow when the evaluation point is in an area of low
density. A value of 0.5 is default and generally recommended.

Value

Marginal
A matrix with two columns of marginal accuracy and consistency per cut score
(and simultaneous if multiple cutscores are given)

Conditional
A list of two matrixes, one for conditional accuracy and one for conditional
consistency. Each matrix has one row per evaluation point.

Note

The function pnr() is modified from Ramsay’s (1991) kernel-smoothed response functions, specif-
cically because they occur conditional total score (and not conditional on a latent trait) and the
addition of an adaptive bandwidth (which helps performance when the distribution of total scores
is not normal.)

There is no "D" method of marginalization (as there is for class.Rud and class.Lee). But if
there is a theoretical distribution of total scores, the pnr.out[[2]] can be adjusted to match this
theoretical distribution.

Author(s)

Quinn N. Lathrop
References

Examples

#Simulate simple response data
params <- matrix(c(1,1,1,1,-2,1,0,1,0,0,0,0),4,3)
theta <- rnorm(100)
rdm <- sim(params, theta)
pnr.out <- pnr(rdm)
resultsNP <- Lee.pnr(3, pnr.out)

recursive.raw

Recursive computation of conditional total score

Description
Recursively computes the probabilities of each possible total score conditional on ability.

Usage
recursive.raw(ip, theta, D = 1.7)
gen.rec.raw(Pij, theta.names = NULL)

Arguments
ip Jx3 matrix of item parameters, columns are discrimination, difficulty, and guessing; in that order.
theta Vector of abilities or points to condition on.
D The scaling constant for the IRT parameters, defaults to 1.7, alternatively often set to 1.
Pij Either:
(1) an NxJ matrix of probabilities of correct response, where each row corresponds to the respective element in theta and each column represents an item (as in the result of irf()$f)
or
(2) an NxMxJ array of probabilities. Each slice of the array represents an item. Within a slice, each row corresponds to the respective element in theta and each
column represents a response category from 0, 1, ..., M. At a minimum, M=1, in which case the array is Nx2xJ and represents the dichotomous item case.

theta.names Optional vector to use as row.names in the output matrix. Should correspond to the first dimension of Pij

Value

A matrix of theta points by possible total score 0,1,...,J.

Note

As described in Huynh 1990.

If the test is mixed format (some dichotomous, some polynominal items), to use gen.rec.raw(), Pij must be of an appropriate size for the item with the most response categories. The response categories that do not appear in other items can be filled with zeros. Note also that the function assumes response categories are scored as 0,1,2,3,...,M

Author(s)

Quinn Lathrop

Examples

theta <- c(-1,0, 1)
params <- matrix(c(1,1,1,1,-2,1,0,1,0,0,0,0),4,3)

#using IRT model and item parameters
rec.mat <- recursive.raw(params, theta)

#using user supplied probability array
Pij.flat <- irf(params, theta)$f

#through matrix input
rec.mat2 <- gen.rec.raw(Pij.flat, theta)

#through array input (this can be generalized to polytomous tests)
Pij.array <- array(NA, dim = c(length(theta), 2, nrow(params)))
Pij.array[,1] <- 1 - Pij.flat #P(X_j = 0 | theta_i)
Pij.array[,2] <- Pij.flat #P(X_j = 1 | theta_i)

rec.mat3 <- gen.rec.raw(Pij.array, theta)

#same results
max(c(rec.mat-rec.mat3, rec.mat2-rec.mat3))
Useful IRT Functions

A collection of useful IRT functions.

Description

Modified from the package irtoys.

Usage

\begin{verbatim}
   iif(ip, x, D = 1.7)
   irf(ip, x, D = 1.7)
   MLE(resp, ip, D = 1.7, min= -4, max = 4)
   normal.qu(n = 15, lower = -4, upper = 4, mu = 0, sigma = 1)
   SEM(ip, x, D = 1.7)
   sim(ip, x, D = 1.7)
   tif(ip, x, D = 1.7)
\end{verbatim}

Arguments

\begin{itemize}
   \item \textit{ip} \hspace{1cm} A Jx3 matrix of item parameters. Columns are discrimination, difficulty, and guessing
   \item \textit{x} \hspace{1cm} Vector of theta points
   \item \textit{resp} \hspace{1cm} Response data matrix, subjects by items
   \item \textit{min, max} \hspace{1cm} MLE is undefined for perfect scores. These parameters define the range in which to search for the MLE, if the score is perfect, the min or max will be returned.
   \item \textit{n} \hspace{1cm} Number of quadrature points wanted
   \item \textit{lower, upper} \hspace{1cm} Range of points wanted
   \item \textit{mu, sigma} \hspace{1cm} The normal distribution from which points and weights are taken
   \item \textit{D} \hspace{1cm} The scaling constant for the IRT parameters, defaults to 1.7, alternatively often set to 1.
\end{itemize}

Details

iif gives item information, irf gives item response function, MLE returns maximum likelihood estimates of theta (perfect scores get \pm 4), normal.qu returns a list length 2 of normal quadrature points and weights, SEM gives the standard error of measurement at the given ability points, sim returns simulated response matrix, tif gives the test information function.

Author(s)

Quinn N. Lathrop

References

Examples

```r
params<-matrix(c(1,1,1,1,-2,1,0,1,0,0,0,0),4,3)
rdm<-sim(params, rnorm(100))

theta.hat <- MLE(rdm, params)
theta.se <- SEM(rdm, params)

## transform a cut score of theta = 0 to the expected true score scale

t.cut <- 0
x.cut <- sum(irf(params, t.cut)$f)
```
Index

* IRT
  Useful IRT Functions, 12
* -Classification
  cacIRT-package, 2
* -IRT
  cacIRT-package, 2
class.Lee, 3
class.Rud, 5
Lee.poly, 6
Nonparametric Approach to CA and CC, 8
recursive.raw, 10
* -Item Response Theory
  cacIRT-package, 2
cacIRT (cacIRT-package), 2
cacIRT-package, 2
class.Lee, 3
class.Rud, 5
gener.rec.raw (recursive.raw), 10
iif (Useful IRT Functions), 12
irf (Useful IRT Functions), 12
Lee.D (class.Lee), 3
Lee.P (class.Lee), 3
Lee.pnr (Nonparametric Approach to CA and CC), 8
Lee.poly, 6
MLE (Useful IRT Functions), 12
Nonparametric Approach to CA and CC, 8
normal.qu (Useful IRT Functions), 12
pnr (Nonparametric Approach to CA and CC), 8
recursive.raw, 10
Rud.D (class.Rud), 5
Rud.P (class.Rud), 5
SEM (Useful IRT Functions), 12
sim (Useful IRT Functions), 12
tif (Useful IRT Functions), 12
Useful IRT Functions, 12