Package ‘cdcatR’

March 4, 2019

Title Congitive Diagnostic Computerized Adaptive Testing in R

Version 0.9

Date 2019-3-4

Author Miguel A. Sorrel [aut, cre], Francisco J. Abad [aut]

Maintainer Miguel A. Sorrel <miguel.sorrel@uam.es>

Description This package holds functions for conducting CD-CAT applications.

Depends R (>= 3.4.0), GDINA (>= 2.2.0), ggplot2, cowplot

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1.9000

R topics documented:

`att.plot` *Create plots for attribute mastery estimates*

Description

Create plots for attribute mastery estimates (X: item position, Y: mastery probability).

Usage

```
att.plot(cdcat.obj, i, ...)
```

Arguments

`cdcat.obj`	An object of class `cdcat`.
`i`	examinee to be plotted.

Value

`att.plot` creates a plot.

`cdcat` *Cognitively based computerized adaptive test application*

Description

`cdcat` conducts a CD-CAT application for a given dataset. Items are selected according to the general discrimination index (GDI; de la Torre & Chiu, 2016; Kaplan, de la Torre, & Barrada, 2016). The next item to be selected by the adaptive algorithm is the one with the highest GDI.

Usage

```
cdcat(GDINA.obj, dat = NULL, MAXJ = 20, FIXED.LENGTH = TRUE,
  att.prior = NULL, post.initial = NULL, max.cut = 0.8,
  i.print = 250, ...)
```

Arguments

`GDINA.obj`	Calibrated item bank.
`dat`	Dataset to be analyzed (if is.null(dat) then dat <- GDINA.obj$options$dat) (i.e., post-hoc CD-CAT simulation).
`MAXJ`	Maximum number of items to be applied. Default is 20.
`FIXED.LENGTH`	Fixed CAT-legnth (TRUE) or fixed-precision (FALSE). Default is TRUE.
`att.prior`	Prior distribution for MAP/EAP estimates.
`post.initial`	Prior distribution for GDI.
`max.cut`	Cutoff for fixed-precision (posterior pattern > max.cut). Default is .80.
`i.print`	Print examinee information. Default is 250.

Value

`cdcat` returns an object of class `cdcat`.

References

de la Torre, J., & Chiu, C. Y. (2016). General Method of Empirical Q-matrix Validation. *Psychometrika, 81*, 253-273.

Kaplan, M., de la Torre, J., & Barrada, J. R. (2015). New item selection methods for cognitive diagnosis computerized adaptive testing. *Applied Psychological Measurement, 39*, 167-188.

Examples

```
####################################
#       Example 1.                 #
#     CD-CAT simulation for a      #
#     GDINA obj                    #
####################################

#--------Data generation --------#
Q <- sim180GDINA$simQ
K <- ncol(Q)
dat <- sim180GDINA$simdat
att <- sim180GDINA$simalpha

#--------Model estimation --------#
fit <- GDINA(dat = dat, Q = Q, verbose = 0)

#--------CD-CAT --------#
res.FIXJ <- cdcat(dat = fit$options$dat,
                 GDINA.obj = fit,
                 FIXED.LENGTH = TRUE)
res.VARJ <- cdcat(dat = fit$options$dat,
                 GDINA.obj = fit,
                 FIXED.LENGTH = FALSE)

#--------Results --------#
res.FIXJ$est[[1]] # estimates for the first examinee (fixed-length)
res.VARJ$est[[1]] # estimates for the first examinee (fixed-precision)
att.plot(res.FIXJ, i = 1) # plot for estimates for the first examinee (fixed-length)
att.plot(res.VARJ, i = 1) # plot for estimates for the first examinee (fixed-length)
# FIXJ summary
res.FIXJ.sum.real <- cdcat.summary(cdcat.obj = res.FIXJ,
                                  alpha = att) # vs. real accuracy
res.FIXJ.sum.real$recovery$plotPCV
res.FIXJ.sum.real$recovery$plotPCA
# VARJ summary
res.VARJ.sum.post <- cdcat.summary(cdcat.obj = res.VARJ, alpha = att)
res.VARJ.sum.post$CATlength$stats
res.VARJ.sum.post$CATlength$plot
res.VARJ.sum.post$recovery
# Post-hoc CAT simulation (only if dat is fit$options$dat)
att.J <- personparm(fit, "MAP")[, -(K+1)]
class.J <- ClassRate(att, att.J) # upper-limit for accuracy
res.FIXJ.sum.post <- cdcat.summary(cdcat.obj = res.FIXJ, alpha = att.J)
res.FIXJ.sum.post$recovery$plotPCV + geom_hline(yintercept = class.J$PCV[K], color = "red")
res.FIXJ.sum.post$recovery$plotPCA + geom_hline(yintercept = class.J$PCA, color = "red")

####################################
# Example 2. #
# CD-CAT simulation for #
# multiple GDINA objs and #
# comparison of performance on a
# validation sample #
####################################

#--------Data --------#
Q <- sim155complex$simQ
```

```
K <- ncol(Q)
parm <- sim155complex$simcatprob.parm
dat.c <- sim155complex$simdat.c
att.c <- sim155complex$simalpha.c
dat.v <- sim155complex$simdat.v
att.v <- sim155complex$simalpha.v

#--------(multiple) Model estimation --------#
fitTRUE <- GDINA(dat = dat.c, Q = Q, catprob.parm = parm, control = list(maxitr = 0), verbose = 0)
fitGDINA <- GDINA(dat = dat.c, Q = Q, verbose = 0)
fitDINA <- GDINA(dat = dat.c, Q = Q, model = "DINA", verbose = 0)
fitDINO <- GDINA(dat = dat.c, Q = Q, model = "DINO", verbose = 0)
fitACDM <- GDINA(dat = dat.c, Q = Q, model = "ACDM", verbose = 0)
LR2 <- modelcomp(GDINA.obj = fitGDINA, method = "LR",
                LR.args = list(LR.approx = TRUE), models = c("DINA", "DINO","ACDM"))
alpha.level <- 0.05
model <- apply(LR2$pvalues, 1, function(x) {
 if (max(x, na.rm = TRUE) > alpha.level/(sum(rowSums(Q)>1)*3)) {
   which.max(x)}
 else {
   return(0)}})
models <- rep(0, nrow(Q))
models[which(rowSums(Q) != 1)] <- model
models <- models
fitLR2 <- GDINA(dat = dat.c, Q = Q, model = models, verbose = 0)

#--------CD-CAT --------#
fit.l <- list(fitTRUE, fitGDINA, fitDINA, fitDINO, fitACDM, fitLR2)
res.FIXJ.l <- res.VARJ.l <- list()
for(mm in 1:length(fit.l)) {
 fit <- fit.l[[mm]]
 res.FIXJ.l[[mm]] <- cdcat(dat = dat.v,
                          GDINA.obj = fit,
                        FIXED.LENGTH = TRUE)
 res.VARJ.l[[mm]] <- cdcat(dat = dat.v,
                        GDINA.obj = fit,
                          FIXED.LENGTH = FALSE)
}

#--------Results --------#
fitbest <- GDINA(dat = dat.v, Q = Q, catprob.parm = parm, control = list(maxitr = 1), verbose = 0)
fitbest.acc <- personparm(fitbest, "MAP")[, -(K+1)]
class.J <- ClassRate(att.v, fitbest.acc) # upper-limit for accuracy
# FIXJ comparison
res.FIXJ.sum.post.comp <- cdcat.comp(cdcat.obj.l = res.FIXJ.l, alpha = att.v)
res.FIXJ.sum.post.comp$PCVcomp + geom_hline(yintercept = class.J$PCV[K], color = "red")
res.FIXJ.sum.post.comp$PCAmcomp + geom_hline(yintercept = class.J$PCA, color = "red")
# VARJ comparison
res.VARJ.sum.post.comp <- cdcat.comp(cdcat.obj.l = res.VARJ.l, alpha = att.v)
res.VARJ.sum.post.comp$stats
res.VARJ.sum.post.comp$plots
res.VARJ.sum.post.comp$recovery
```

`cdcat.comp` *Comparison of multiple cdcat objects*

Description

This function compares differet `cdcat` objects in terms of classification accuracy (FIXED.LENGTH == TRUE) and/or CAT lenght FIXED.LENGTH == FALSE).

Usage

```
cdcat.comp(cdcat.obj.l, alpha, ...)
```

Arguments

`cdcat.obj.l`	List of `cdcat` objects to be compared.
`alpha`	N x K matrix with the attribute patterns to be compared to the `cdcat` results.

Value

`cdcat.comp` returns an object of class `cdcat.comp`.

`cdcat.summary` *Summary information for a cdcat.object*

Description

This function provides classification accuracy (FIXED.LENGTH == TRUE) and/or CAT lenght FIXED.LENGTH == FALSE) results for cdcat object.

Usage

```
cdcat.summary(cdcat.obj, alpha, ...)
```

Arguments

`cdcat.obj`	`cdcat` results.
`alpha`	N x K matrix with the attribute patterns to be compared to the `cdcat` results.

Value

`cdcat.summary` returns an object of class `cdcat.summary`.

gen.itembank *Item bank generation*

Description

This function can be used to generate an item bank. The user can provide a Q-matrix or create one defining the number of times each attribute should be measured and the q-vector complexity (e.g., number of attributes that a q-vector can measure). Item parameters are sampled from a uniform distribution with mean = *IQ* and variance = *VAR*.

Usage

```
gen.itembank(Q = NULL, minJ.K = NULL, complexity = NULL, IQ, VAR,
  model = NULL, ...)
```

Arguments

Q	Q-matrix.
minJ.K	Vector indicating the minimum number of items measuring each attribute.
complexity	Vector indicating the maximum number of attributes being measured by an item in each column of Q. At this moment maximum is 4.
IQ	Item discrimination (mean for the uniform distribution). IQ = $P(\mathbf{1})$ - $P(\mathbf{0})$ (Sorrel, Abad, Olea, de la Torre, and Barrada, 2017).
VAR	Item discrimination (variance for the uniform distribution).
model	Vector indicating the model-item correspondnce (0 = one-attribute item, 1 = DINA, 2 = DINO, 3 = A-CDM.

Value

gen.itembank returns an object of class gen.Item.Bank.

References

Kaplan, M., de la Torre, J., & Barrada, J. R. (2015). New item selection methods for cognitive diagnosis computerized adaptive testing. *Applied Psychological Measurement, 39*, 167-188.

Sorrel, M. A., Abad, F. J., Olea, J., de la Torre, J., & Barrada, J. R. (2017). Inferential item-fit evaluation in cognitive diagnosis modeling. *Applied Psychological Measurement, 41*, 614-631.

Examples

```
####################################
#        Example 1.                #
#     Q and model are provided     #
####################################
Q <- sim30GDINA$simQ
model <- rep(1, each = nrow(Q))
IQ <- .70   # P(1), IQ = Low item quality in Kaplan, de la Torre & Barrada (2015)
VAR <- 0.10 # High variance in Kaplan et al. (2015)
bank <- gen.itembank(Q = Q, IQ = IQ, VAR = VAR, model = model)

####################################
```

```
#       Example 2.              #
#   Q and model not are provided   #
####################################

minJ.K  <- c(50, 50, 50)
complexity <- c(3, 3, 3)
IQ <- .70   # P(1), IQ = Low item quality in Kaplan, de la Torre & Barrada (2015)
VAR <- 0.10 # High variance in Kaplan et al. (2015)
bank <- gen.itembank(minJ.K = minJ.K, complexity = complexity, IQ = IQ, VAR = VAR)
```

---

sim155complex *Simulated data (155 items, a combination of DINA, DINO, and A-CDM models)*

Description

Simulated data, Q-matrix and item parameters for a 155-item bank with 5 attributes.

Usage

```
sim155complex
```

Format

A list with components:

- `simQ` Artificial Q-matrix. Q-matrix structure is complex (items measure up to four attributes and only 5 of them are one-attribute items
- `simcatprob.parm` Artificial item parameters (probability of success for each latent group). Items 1-60 are DINA items, items 61-120 are DINO items, and items 121-180 are A-CDM items
- `simdat.c` Calibration sample dataset. Simulated responses of 500 examinees
- `simalpha.c` Calibration sample alpha patterns. simulated attribute patterns of 500 examinees)
- `simdat.v` Validation sample dataset. Simulated responses of 500 examinees
- `simalpha.v` Validation sample alpha patterns. simulated attribute patterns of 500 examinees)

---

sim155simple *Simulated data (155 items, a combination of DINA, DINO, and A-CDM models)*

Description

Simulated data, Q-matrix and item parameters for a 155-item bank with 5 attributes.

Usage

```
sim155simple
```

Format

A list with components:

`simQ` Artificial Q-matrix. Q-matrix structure is simple (items measure up to three attributes and 35 of them are one-attribute items

`simcatprob.parm` Artificial item parameters (probability of success for each latent group). Items 1-60 are DINA items, items 61-120 are DINO items, and items 121-180 are A-CDM items

`simdat.c` Calibration sample dataset. Simulated responses of 500 examinees

`simalpha.c` Calibration sample alpha patterns. simulated attribute patterns of 500 examinees)

`simdat.v` Validation sample dataset. Simulated responses of 500 examinees

`simalpha.v` Validation sample alpha patterns. simulated attribute patterns of 500 examinees)

`sim180DINA` *Simulated data (180 items, DINA model)*

Description

Simulated data, Q-matrix and item parameters for a 180-item bank with 5 attributes.

Simulated data, Q-matrix and item parameters for a 180-item bank with 5 attributes.

Usage

```
sim180DINA

sim180DINA
```

Format

A list with components:

`simdat` Simulated responses of 500 examinees

`simQ` Artificial Q-matrix

`simcatprob.parm` Artificial item parameters (probability of success for each latent group). All items are DINA items

`simalpha` Simulated attribute patterns of 500 examinees)

Index