

Package ‘cdCAT’

June 15, 2026

Type Package

Title Computerized Adaptive Testing with Cognitive Diagnostic Models

Version 0.1.0

Description A session-based engine for cognitive diagnostic computerized adaptive testing (CD-CAT), the application of adaptive testing to cognitive diagnosis models. Three models are supported: the deterministic inputs, noisy “and” gate (DINA), the deterministic inputs, noisy “or” gate (DINO), and the generalized DINA (GDINA) model. Item selection criteria include Kullback-Leibler (KL) information, posterior-weighted Kullback-Leibler (PWKL), modified posterior-weighted Kullback-Leibler (MPWKL), and Shannon entropy (SHE). Latent attribute profiles are estimated by maximum likelihood estimation (MLE), maximum a posteriori (MAP), or expected a posteriori (EAP). Content balancing, item exposure control, and shadow testing are configurable through constraint functions. The implemented methods follow Cheng (2009) [doi:10.1007/s11336-009-9123-2](https://doi.org/10.1007/s11336-009-9123-2) and de la Torre (2011) [doi:10.1007/s11336-011-9207-7](https://doi.org/10.1007/s11336-011-9207-7). Designed for real-time, item-by-item adaptive applications.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.1.0)

Imports R6

Suggests testthat (>= 3.0.0), knitr, rmarkdown, covr, GDINA, usethis

Config/testthat/edition 3

URL <https://github.com/thiagofmiranda/cdCAT>

BugReports <https://github.com/thiagofmiranda/cdCAT/issues>

VignetteBuilder knitr

LazyData true

NeedsCompilation no

Author Thiago Miranda [aut, cre],
Bruno Silvestre [aut]

Maintainer Thiago Miranda <thiagofm.pa@gmail.com>

Repository CRAN

Date/Publication 2026-06-15 13:40:08 UTC

Contents

apply_content_balancing	2
apply_exposure_control	4
apply_randomesque	5
apply_sympson_hetter	6
build_skill_patterns	6
CdcatSession	7
cdcat_items	11
cdcat_prior	12
cdcat_sim	13
check_stopping	15
estimate_alpha	17
get_prob_matrix	18
KL_criteria	18
MPWKL_criteria	19
PWKL_criteria	20
select_next_item	21
SHE_criteria	23
Index	25

apply_content_balancing

Apply content balancing to candidate items

Description

Filters the candidate item pool so that the next item is drawn from the content domain most under-represented relative to the target blueprint (Kingsbury & Zara, 1991).

Usage

```
apply_content_balancing(candidate_items, administered, content, content_prop)
```

Arguments

candidate_items	Integer vector of indices of items eligible for selection (i.e. not yet administered).
administered	Integer vector of indices of items already administered. Length 0 is valid (no items administered yet).
content	Character vector of length J (total item bank size) giving the content domain label for each item. Example: c("algebra", "algebra", "geometry", "geometry").
content_prop	Named numeric vector of target proportions for each domain. Names must match the unique values in content. Values must be non-negative and sum to 1. Example: c(algebra = 0.6, geometry = 0.4).

Details

The gap for each domain is defined as:

$$\text{gap}_d = \text{prop_target}_d - \text{prop_observed}_d$$

The domain with the largest gap is selected, and only items from that domain are returned. If no candidate items belong to the target domain, the full candidate pool is returned as a safe fallback (no items are left un-selectable).

Value

Integer vector of candidate item indices, restricted to the most under-represented domain (or the full candidate_items if no balancing is needed or no candidates belong to that domain).

References

Kingsbury, G. G., & Zara, A. R. (1991). A comparison of procedures for content-sensitive item selection in computerized adaptive testing. *Applied Measurement in Education*, 4(3), 241–261. [doi:10.1207/s15324818ame0403_4](https://doi.org/10.1207/s15324818ame0403_4)

See Also

[select_next_item\(\)](#), [Cdcatsession](#)

Examples

```
content      <- c("A", "A", "A", "B", "B", "B")
content_prop <- c(A = 0.5, B = 0.5)

# No items administered yet -- returns all candidates unchanged
apply_content_balancing(1:6, integer(0), content, content_prop)

# After two "A" items: gap favours "B"
apply_content_balancing(3:6, c(1L, 2L), content, content_prop)
```

 apply_exposure_control

Apply exposure control to item selection

Description

Selects the next item from the candidate pool according to the exposure control regime encoded in exposure.

Usage

```
apply_exposure_control(scores, available, exposure, n_administered)
```

Arguments

scores	Numeric vector of criterion scores for the candidate items (same length and order as available).
available	Integer vector of candidate item indices (items eligible for selection at this step).
exposure	Numeric vector of length J (total items) with exposure parameters, or NULL for no control. See Details.
n_administered	Integer. Number of items already administered (used to determine position k in the Randomesque regime).

Details

Two regimes are supported and detected automatically from the values of exposure:

Sympson-Hetter (Sympson & Hetter, 1985) – triggered when **all** values in exposure are in $[0, 1]$. Each item has an acceptance probability equal to its exposure value. Items are visited in descending score order; the first one that passes the random draw is selected. Values near 1 are almost always selected; values near 0 are rarely selected.

Randomesque – triggered when **all** values in exposure are ≥ 1 . At position k in the test (i.e. when selecting the k -th item), the top exposure[k] candidates by score are collected and one is drawn at random. exposure[k] = 1 is equivalent to greedy selection.

When exposure is NULL, greedy selection (which.max(scores)) is used, equivalent to no exposure control.

Value

Integer scalar – global index of the selected item.

References

Sympson, J. B., & Hetter, R. D. (1985). *Controlling item-exposure rates in computerized adaptive testing*. Proceedings of the 27th annual meeting of the Military Testing Association (pp. 973–977).

See Also

[apply_sympson_hetter\(\)](#), [apply_randomesque\(\)](#), [select_next_item\(\)](#), [Cdcatsession](#)

Examples

```
scores <- c(0.8, 0.5, 0.3, 0.9)
available <- c(2L, 4L, 6L, 8L)

# Greedy (no exposure control)
apply_exposure_control(scores, available, exposure = NULL, n_administered = 0L)

# Sympson-Hetter: item 8 (score 0.9) has probability 0.2 of being accepted
set.seed(1)
exposure_sh <- rep(0.8, 10)
exposure_sh[8] <- 0.2
apply_exposure_control(scores, available, exposure_sh, n_administered = 2L)

# Randomesque: at position 3, draw from top-2 candidates
exposure_rq <- rep(1, 10)
exposure_rq[3] <- 2
set.seed(1)
apply_exposure_control(scores, available, exposure_rq, n_administered = 2L)
```

apply_randomesque	<i>Randomesque exposure control</i>
-------------------	-------------------------------------

Description

Selects an item by drawing uniformly at random from the top-n candidates by score. When $n = 1$ the result is identical to greedy (deterministic) selection.

Usage

```
apply_randomesque(scores, available, n)
```

Arguments

scores	Numeric vector of scores for the candidate items.
available	Integer vector of global item indices corresponding to scores.
n	Integer. Number of top-scoring candidates to include in the random draw. Capped at $\text{length}(\text{scores})$ if larger.

Value

Integer scalar – global index of the selected item.

See Also

[apply_exposure_control\(\)](#)

apply_sympson_hetter *Sympson-Hetter exposure control*

Description

Selects an item by iterating over candidates in descending score order and accepting the first one that passes a Bernoulli draw with probability $p[\text{item}]$ (Sympson & Hetter, 1985).

Usage

```
apply_sympson_hetter(scores, available, p)
```

Arguments

scores	Numeric vector of scores for the candidate items.
available	Integer vector of global item indices corresponding to scores.
p	Numeric vector of acceptance probabilities in $[0, 1]$, one per item in the bank (length J). The function extracts probabilities for available items internally.

Value

Integer scalar – global index of the selected item.

References

Sympson, J. B., & Hetter, R. D. (1985). *Controlling item-exposure rates in computerized adaptive testing*. Proceedings of the 27th annual meeting of the Military Testing Association (pp. 973–977).

See Also

[apply_exposure_control\(\)](#)

build_skill_patterns *Build skill patterns matrix*

Description

Generates all 2^K binary skill profiles for K attributes.

Usage

```
build_skill_patterns(K)
```

Arguments

K	Integer. Number of attributes.
---	--------------------------------

Value

A matrix of 2^K rows and K columns.

Cdcatsession	<i>CD-CAT Session Object</i>
--------------	------------------------------

Description

CD-CAT Session Object

CD-CAT Session Object

Details

An R6 class implementing the adaptive testing cycle for a single cognitive diagnosis computerized adaptive testing (CD-CAT) session.

This class orchestrates the sequential CD-CAT procedure:

1. Initialize the posterior distribution over latent profiles;
2. Select the next item according to a specified selection criterion;
3. Observe the response and update the posterior;
4. Evaluate stopping criteria;
5. Repeat until termination.

The adaptive algorithm assumes:

- Binary item responses;
- Conditional independence of item responses given the latent profile;
- A calibrated item bank defined by a Q-matrix and fixed item parameters;
- A specified estimation method (MLE, MAP, or EAP);
- A specified item selection criterion (e.g., KL, PWKL, MPWKL, SHE).

The session object maintains the full state of the adaptive test, including responses, administered items, posterior distribution, and stopping status.

Implementation details

Each call to `next_item()` evaluates stopping criteria, computes item selection (with full diagnostic details), and caches the result in `private$pending`. Each call to `update(item, response)` records the response, re-estimates the attribute profile, and appends a complete step record to `self$history`.

Per-step history

The history field stores one named list per administered item. Use `session$history_df()` to convert to a flat data.frame. Each record contains:

`step` Integer. 1-based position in the test.

item Integer. Item index administered.
 response Integer. 0 or 1.
 candidate_items Integer vector. Items available after content balancing filter.
 criterion_scores Named numeric vector. Criterion score per candidate item (NULL for SEQ/RANDOM).
 item_pre_exposure Integer. Item that would have been chosen without exposure control (NA in shadow/non-adaptive mode).
 exposure_redirected Logical. Whether exposure control redirected the selection.
 target_domain Character. Content domain targeted by balancing at this step (NULL if inactive).
 domain_gap Named numeric. Gap (target - observed) per domain at this step (NULL if inactive).
 stop_check List with stop (logical) and reason (character) from the stopping evaluation before this item.
 posterior Numeric vector of length 2^K . Posterior after this response.
 alpha_hat Integer vector of length K. Estimated profile.
 alpha_hat_index Integer. Row index of alpha_hat in the skill patterns matrix.
 mastery_marginal Named numeric of length K. $P(\alpha_k = 1)$ for each attribute after this response.
 timestamp POSIXct. Time at which update() was called.
 response_time Numeric. Seconds elapsed since the previous update() call (or session start for the first item).

Public fields

items A cdcats_items object.
 method Estimation method ("MAP", "MLE", or "EAP").
 criterion Item selection criterion.
 min_items Minimum items before stopping is allowed.
 max_items Maximum items allowed.
 threshold Posterior threshold for attribute classification.
 prior Prior probability vector over skill profiles, or NULL.
 content Character vector of domain labels per item, or NULL.
 content_prop Named numeric vector of target domain proportions.
 exposure Numeric exposure vector per item, or NULL.
 constr_fun Shadow CAT constraint function, or NULL.
 initial_profile Binary integer vector of length K for first-item anchor override, or NULL.
 start_item Integer or character. Method for selecting the first item.
 responses Numeric vector of responses (NA = not administered).
 administered Integer vector of administered item indices.
 est Latest cdcats_est estimation result.
 stop_reason Character stopping reason, or NULL.
 history List of per-step records (one element per administered item). See Details for field descriptions.

Methods

Public methods:

- `Cdcatsession$new()`
- `Cdcatsession$next_item()`
- `Cdcatsession$update()`
- `Cdcatsession$history_df()`
- `Cdcatsession$result()`
- `Cdcatsession$print()`
- `Cdcatsession$clone()`

Method `new()`: Create a new CD-CAT session.

Usage:

```
Cdcatsession$new(
  items,
  method = "MAP",
  criterion = "PWKL",
  min_items = 1L,
  max_items = 20L,
  threshold = 0.8,
  prior = NULL,
  content = NULL,
  content_prop = NULL,
  exposure = NULL,
  constr_fun = NULL,
  initial_profile = NULL,
  start_item = NULL
)
```

Arguments:

`items` A `cdcat_items` object.

`method` Estimation method. Default "MAP".

`criterion` Item selection criterion. Default "PWKL".

`min_items` Minimum items. Default 1.

`max_items` Maximum items. Default 20.

`threshold` Classification threshold. Default 0.8.

`prior` Prior probability vector (length 2^K) or output of `cdcat_prior()`. NULL uses uniform prior.

`content` Character vector of length J with domain labels. NULL disables content balancing.

`content_prop` Named numeric vector of target domain proportions. NULL disables content balancing.

`exposure` Numeric vector of length J. Values in $[0, 1]$ -> Simpson-Hetter; values ≥ 1 -> Randomesque. NULL disables.

`constr_fun` Constraint function for shadow CAT, or NULL.

`initial_profile` Binary integer vector of length K. When provided, used as the anchor profile for KL/PWKL on the first item selection only; does not affect the posterior.

`start_item` Integer or character. Specifies how the first item is selected. An integer administers that specific item index; "random" selects randomly; "seq" selects the first available item; a criterion name ("KL", "PWKL", "MPWKL", "SHE") applies that criterion for the first selection only. NULL (default) applies the main criterion from the very first item, including content balancing, exposure control, and shadow CAT constraints.

Method `next_item()`: Select the next item to administer.

Evaluates stopping criteria, computes item selection, and caches selection details internally for `update()` to incorporate into the history record.

Usage:

```
Cdcatsession$next_item()
```

Returns: Integer index of the next item, or 0L if stopped.

Method `update()`: Register a response and update session state.

Records the response, re-estimates the attribute profile, and appends a complete step record to `self$history`.

Usage:

```
Cdcatsession$update(item, response)
```

Arguments:

`item` Integer. Index of the administered item.

`response` Numeric. 1 (correct) or 0 (incorrect).

Method `history_df()`: Convert the history list to a flat data.frame.

Vector-valued fields (`posterior`, `mastery_marginal`, `alpha_hat`, `criterion_scores`, `candidate_items`, `domain_gap`) are kept as list-columns.

Usage:

```
Cdcatsession$history_df()
```

Returns: A data.frame with one row per administered item, or an empty data.frame if no items have been administered.

Method `result()`: Extract final session results.

Usage:

```
Cdcatsession$result()
```

Returns: Named list with estimation results and session metadata.

Method `print()`: Print a summary of the session state.

Usage:

```
Cdcatsession$print(...)
```

Arguments:

... Ignored.

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
Cdcatsession$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

cdcat_items

*Create a cdCAT Item Bank***Description**

Constructs and validates the item bank object used in cognitive diagnosis computerized adaptive testing (CD-CAT) sessions.

Usage

```
cdcat_items(q_matrix, model, slip = NULL, guess = NULL, gdina_params = NULL)
```

Arguments

q_matrix	A binary matrix of items (rows) by attributes (columns).
model	A string: "DINA", "DINO", or "GDINA".
slip	Numeric vector of slip parameters (required for DINA/DINO).
guess	Numeric vector of guess parameters (required for DINA/DINO).
gdina_params	Named list of item parameters (required for GDINA).

Details

The item bank is defined by a Q-matrix and model-specific item parameters.

The Q-matrix is a binary matrix where:

$$q_{jk} = \begin{cases} 1 & \text{if item } j \text{ requires attribute } k, \\ 0 & \text{otherwise.} \end{cases}$$

Three cognitive diagnosis models are supported:

DINA model (Junker & Sijtsma, 2001):

$$P(X_j = 1|\alpha) = (1 - s_j)^{\eta_j(\alpha)} g_j^{1-\eta_j(\alpha)},$$

where s_j is the slip parameter and g_j is the guess parameter.

DINO model shares the same functional form but uses a compensatory OR-type mastery indicator.

GDINA model (de la Torre, 2011) is a saturated model allowing arbitrary interaction effects among required attributes.

All models assume:

- Binary item responses;
- Conditional independence of item responses given the latent profile;
- Fixed item parameters during the adaptive session.

Value

A list of class `cdcat_items` containing the Q-matrix, item parameters, and model specification.

References

Junker, B. W., & Sijtsma, K. (2001). Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement*, 25(3), 258–272. doi:10.1177/01466210122032064

de la Torre, J. (2011). The generalized DINA model framework. *Psychometrika*, 76(2), 179–199. doi:10.1007/s1133601192077

cdcat_prior

Build a named prior distribution over skill profiles

Description

Constructs and validates a prior probability vector over all 2^K latent skill profiles, using pattern names as keys. The pattern names follow the same ordering as `build_skill_patterns()`: `expand.grid` with the first attribute varying fastest.

Usage

```
cdcat_prior(..., K = NULL, normalize = TRUE)
```

Arguments

...	Named numeric values. Each name must be a binary string of length K (e.g. "00", "01", "10", "11" for $K = 2$). All 2^K patterns must be supplied.
K	Integer. Number of attributes. Inferred from pattern length if not supplied.
normalize	Logical. If TRUE (default), the vector is automatically normalized to sum to 1 with a warning when needed.

Value

A named numeric vector of length 2^K whose entries sum to 1, ordered consistently with `build_skill_patterns()`.

See Also

[estimate_alpha\(\)](#), [Cdcatsession](#)

Examples

```
# K = 2: four profiles -- "00", "10", "01", "11"
prior <- cdcatsession("00" = 0.4, "10" = 0.2, "01" = 0.2, "11" = 0.2)

# Use in a session
Q <- matrix(c(1,0, 0,1, 1,1), nrow = 3, ncol = 2, byrow = TRUE)
items <- cdcatsession(Q, "DINA", slip = c(0.1,0.1,0.1), guess = c(0.2,0.2,0.2))
session <- Cdcatsession$new(items, prior = prior)
```

cdcat_sim

*Simulated CD-CAT datasets***Description**

A named list of 16 simulated datasets for Cognitive Diagnostic Computerized Adaptive Testing (CD-CAT) research. The 16 elements cover all combinations of:

- **K** (number of attributes): 3 or 5
- **Model**: DINA or GDINA
- **Quality**: HD-LV, HD-HV, LD-LV, or LD-HV (see *Item quality* section below)

Usage

cdcat_sim

Format

A named list with 16 elements. Element names follow the pattern $k\{K\}_{\{MODEL\}}_{\{QUALITY\}}$, e.g. "k3_DINA_HDLV" or "k5_GDINA_LDHV". Each element is itself a list with four components:

Q Integer matrix ($J \times K$). Q-matrix mapping items to attributes. Constructed by replicating each of the $2^K - 1$ non-zero attribute profiles 10 times, yielding $J = 10 \times (2^K - 1)$ items ($K = 3$: $J = 70$; $K = 5$: $J = 310$). Rows are named *item1*, ..., *itemJ*; columns *A1*, ..., *AK*.

alpha Integer matrix ($N \times K$). True latent attribute profiles of the examinees. Constructed by replicating each of the 2^K possible binary profiles 20 times, yielding $N = 20 \times 2^K$ examinees ($K = 3$: $N = 160$; $K = 5$: $N = 640$). Rows are named *examinee1*, ..., *examineeN*; columns *A1*, ..., *AK*.

parameters List of length J . Item parameters whose format depends on the model:

DINA Each element is a list with two scalars: *slip* (probability that a master answers incorrectly) and *guess* (probability that a non-master answers correctly).

GDINA Each element is a named list of 2^{K_j} scalars, where K_j is the number of attributes required by item j . Names are binary strings representing the latent response pattern (e.g. "0", "1" for $K_j = 1$; "00", "10", "01", "11" for $K_j = 2$). Each value is the probability of a correct response given that pattern of required-attribute mastery.

responses Integer matrix ($N \times J$). Dichotomous item responses (0 / 1) simulated from **alpha** and **parameters** using GDINA: :simGDINA().

Item quality

Parameters were sampled from uniform distributions on ranges that reflect four levels of item quality, defined by the probability of a correct response for non-masters (P_0) and masters (P_1):

Code	Label	P_0 range	P_1 range
HDLV	High Discrimination, Low Variability	[0.05, 0.15]	[0.85, 0.95]

HDHV	High Discrimination, High Variability	[0.00, 0.20]	[0.80, 0.999]
LDLV	Low Discrimination, Low Variability	[0.15, 0.25]	[0.75, 0.85]
LDHV	Low Discrimination, High Variability	[0.10, 0.30]	[0.70, 0.90]

For **DINA/DINO**: $\text{slip} = 1 - P_1$ and $\text{guess} = P_0$, independently sampled per item and clipped to $[0.001, 0.999]$.

For **GDINA**: $P_{min} \sim P_0$ and $P_{max} \sim P_1$ are independently sampled per item; the probability of a correct response increases linearly with the count of mastered required attributes: $P_{min} + (m/K_j) \times (P_{max} - P_{min})$, where m is the number of required attributes mastered.

Reproducibility

All datasets were generated with `set.seed(2025)`. The generation script is available in `data-raw/generate_cdcat_sim.R`.

Dataset dimensions

K	J (items)	N (examinees)
3	70	160
5	310	640

Source

Generated with `GDINA::simGDINA()` (Ma & de la Torre, 2020). See `data-raw/generate_cdcat_sim.R` for the full script.

References

Ma, W., & de la Torre, J. (2020). GDINA: An R package for cognitive diagnosis modeling. *Journal of Statistical Software*, 93(14), 1–26. doi:10.18637/jss.v093.i14

Examples

```
# List all available datasets
names(cdcat_sim)

# Access a specific dataset
d <- cdcat_sim[["k3_DINA_HDLV"]]
dim(d$Q)      # 70 x 3
dim(d$alpha)  # 160 x 3
dim(d$responses) # 160 x 70

# Inspect DINA parameters for the first item
d$parameters[[1]]

# Inspect GDINA parameters for the first item
```

```

cdcat_sim[["k3_GDINA_HDLV"]]$parameters[[1]]

# Use in a CD-CAT session
d <- cdcat_sim[["k3_DINA_HDLV"]]
items <- cdcat_items(
  d$d, "DINA",
  slip = sapply(d$parameters, `[`, "slip"),
  guess = sapply(d$parameters, `[`, "guess")
)
session <- Cdcatsession$new(items, criterion = "PWKL", max_items = 10L)

```

check_stopping

Evaluate CD-CAT Stopping Criteria

Description

Evaluates whether the computerized adaptive test should terminate based on the current posterior distribution over latent profiles and operational constraints.

Usage

```

check_stopping(
  est,
  n_administered,
  min_items = 1L,
  max_items = 20L,
  threshold = 0.8
)

```

Arguments

<code>est</code>	A list of class <code>cdcat_est</code> returned by <code>estimate_alpha()</code> .
<code>n_administered</code>	Integer. Number of items already administered.
<code>min_items</code>	Integer. Minimum number of items before stopping is allowed.
<code>max_items</code>	Integer. Maximum number of items allowed.
<code>threshold</code>	Numeric vector of length 1 or 2: <ul style="list-style-type: none"> Length 1: attribute-level posterior threshold τ. Length 2: dual class-level thresholds (τ_1, τ_2).

Details

This function implements three stopping mechanisms within a cognitive diagnosis computerized adaptive testing (CD-CAT) framework:

1. Fixed-length rule

The test stops when the number of administered items reaches `max_items`. This corresponds to the traditional fixed-length stopping rule described in the CD-CAT literature.

2. Attribute-level posterior threshold rule (Tatsuoka, 2002)

Let $\pi_t(\alpha_c)$ denote the posterior distribution over latent profiles after t administered items.

The marginal posterior probability of mastery for attribute k is:

$$P(\alpha_k = 1 | x^{(t)}) = \sum_{c:\alpha_{ck}=1} \pi_t(\alpha_c).$$

The test stops when all attributes are classified with sufficient confidence:

$$P(\alpha_k = 1 | x^{(t)}) \geq \tau \quad \text{or} \quad P(\alpha_k = 1 | x^{(t)}) \leq 1 - \tau \quad \text{for all } k.$$

This rule performs classification at the attribute level rather than at the full latent class level.

3. Dual posterior threshold rule (Hsu, Wang, & Chen, 2013)

Let $\pi_{(1)}$ and $\pi_{(2)}$ denote the largest and second-largest posterior probabilities among all latent profiles.

The test stops when:

$$\pi_{(1)} \geq \tau_1 \quad \text{and} \quad \pi_{(2)} \leq \tau_2,$$

where $\tau_1 > \tau_2$. This conservative rule avoids stopping when two latent profiles remain similarly probable.

In addition to these theoretical criteria, the function enforces:

- A minimum number of administered items (`min_items`);
- A maximum number of administered items (`max_items`).

All posterior computations assume conditional independence of item responses given the latent profile.

Value

A list with:

- `stop`: Logical. Whether the test should stop.
- `reason`: Character string describing the stopping condition.

References

Tatsuoka, C. (2002). Data analytic methods for latent partially ordered classification models. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 51(3), 337–350. doi:10.1111/1467-9876.00272

Hsu, C. L., Wang, W. C., & Chen, S. Y. (2013). Variable-length computerized adaptive testing based on cognitive diagnosis models. *Applied Psychological Measurement*, 37(7), 563–582. doi:10.1177/0146621613488642

estimate_alpha	<i>Estimate Latent Attribute Profiles</i>
----------------	---

Description

Estimates the latent attribute profile given observed item responses under a specified cognitive diagnosis model.

Usage

```
estimate_alpha(responses, items, method = "MAP", prior = NULL)
```

Arguments

responses	Numeric vector of item responses.
items	A cdcats_items object.
method	Estimation method: "MLE", "MAP", or "EAP".
prior	Prior distribution over profiles.

Details

The likelihood for latent profile α_c is:

$$L(\alpha_c) = \prod_{j \in \mathcal{J}_t} P(X_j = x_j | \alpha_c)$$

MLE selects:

$$\hat{\alpha}_{MLE} = \arg \max_c L(\alpha_c)$$

MAP selects:

$$\hat{\alpha}_{MAP} = \arg \max_c L(\alpha_c) \pi(\alpha_c)$$

EAP estimates attribute mastery probabilities:

$$\hat{\alpha}_{EAP,k} = \sum_c \pi_t(\alpha_c) \alpha_{ck}$$

Value

A list of class cdcats_est.

get_prob_matrix	<i>Compute Item Response Probability Matrix</i>
-----------------	---

Description

Computes the matrix of item response probabilities $P(X_j = 1|\alpha)$ for all items and all latent attribute profiles under the specified cognitive diagnosis model.

Usage

```
get_prob_matrix(q_matrix, parameters, skill_patterns, model)
```

Arguments

q_matrix	Binary matrix of items by attributes.
parameters	Model-specific item parameters.
skill_patterns	Matrix of all 2^K attribute profiles.
model	Character string specifying the CDM.

Details

This function computes item response probabilities under three cognitive diagnosis models:

DINA model:

$$P(X_j = 1|\alpha) = (1 - s_j)^{\eta_j(\alpha)} g_j^{1-\eta_j(\alpha)}$$

DINO model: Same functional form as DINA, but with $\eta_j(\alpha) = 1$ if at least one required attribute is mastered.

GDINA model: A saturated model allowing arbitrary interaction effects among required attributes.

Value

A matrix of dimension $J \times 2^K$.

KL_criteria	<i>Kullback-Leibler (KL) Information Criterion for CD-CAT</i>
-------------	---

Description

Computes the Kullback-Leibler (KL) information index for a candidate item under a cognitive diagnosis computerized adaptive testing (CD-CAT) framework.

Usage

```
KL_criteria(item_index, alpha_hat_index, prob_matrix)
```

Arguments

item_index	Integer. Index of the candidate item.
alpha_hat_index	Integer. Index of the current latent profile estimate.
prob_matrix	Numeric matrix of dimension J x 2^K containing item response probabilities P(X=1 alpha).

Details

This criterion selects the item that maximizes the Kullback-Leibler divergence between the item response distribution induced by the current estimated latent profile and the response distributions induced by all alternative latent profiles.

Let $\hat{\alpha}^{(t)}$ denote the current latent profile estimate after t administered items, and let α_c , $c = 1, \dots, 2^K$, denote all possible latent attribute profiles.

For a candidate item h , the KL index is defined as:

$$KL_h(\hat{\alpha}^{(t)}) = \sum_{c=1}^{2^K} \sum_{x=0}^1 P(X_h = x | \hat{\alpha}^{(t)}) \log \frac{P(X_h = x | \hat{\alpha}^{(t)})}{P(X_h = x | \alpha_c)}$$

Thus, the divergence is computed between conditional response distributions for the candidate item under different latent profiles, not between the attribute vectors themselves.

The logarithm is computed using the natural logarithm.

Assumptions:

- Conditional independence of item responses.
- A fixed cognitive diagnosis model defining $P(X = 1 | \alpha)$.
- Equal weighting of all latent profiles.

Computational complexity per candidate item is $O(2^K)$.

Value

Numeric scalar representing the KL information value.

MPWKL_criteria	<i>Modified Posterior-Weighted Kullback-Leibler (MPWKL) Criterion</i>
----------------	---

Description

Computes the symmetric posterior-weighted KL divergence across all pairs of item response distributions induced by latent profiles.

Usage

```
MPWKL_criteria(item_index, prob_matrix, posterior)
```

Arguments

item_index	Integer. Index of the candidate item.
prob_matrix	Numeric matrix of dimension $J \times 2^K$ containing item response probabilities.
posterior	Numeric vector of posterior probabilities.

Details

The MPWKL index for item h is defined as:

$$MPWKL_h = \sum_{d=1}^{2^K} \sum_{c=1}^{2^K} KL_h(\alpha_d || \alpha_c) \pi_t(\alpha_d) \pi_t(\alpha_c)$$

where $KL_h(\alpha_d || \alpha_c)$ denotes the KL divergence between the conditional response distributions of item h under latent profiles α_d and α_c .

This formulation evaluates the expected global discrimination of the item under the full posterior distribution.

Computational complexity per candidate item is $O((2^K)^2)$.

Value

Numeric scalar representing the MPWKL information value.

References

Cheng, Y. (2009). When cognitive diagnosis meets computerized adaptive testing: CD-CAT. *Psychometrika*, 74(4), 619–632. doi:10.1007/s1133600991232

PWKL_criteria

Posterior-Weighted Kullback-Leibler (PWKL) Criterion

Description

Computes the posterior-weighted KL information index for a candidate item in CD-CAT.

Usage

```
PWKL_criteria(item_index, alpha_hat_index, prob_matrix, posterior)
```

Arguments

item_index	Integer. Index of the candidate item.
alpha_hat_index	Integer. Index of the current latent profile estimate.
prob_matrix	Numeric matrix of dimension $J \times 2^K$ containing item response probabilities.
posterior	Numeric vector of posterior probabilities over the 2^K latent profiles.

Details

Unlike the standard KL method, this criterion weights each latent profile by its current posterior probability rather than assuming equally likely latent states.

Let $\pi_t(\alpha_c)$ denote the posterior probability of latent profile α_c after t administered items.

The PWKL index for item h is defined as:

$$PWKL_h(\hat{\alpha}^{(t)}) = \sum_{c=1}^{2^K} KL_h(\hat{\alpha}^{(t)} \parallel \alpha_c) \pi_t(\alpha_c)$$

where $KL_h(\cdot)$ represents the KL divergence between the conditional response distributions induced by two latent profiles.

This weighting reduces early-stage instability and avoids the implicit assumption of uniformly distributed latent states.

Assumptions:

- Correct posterior updating.
- Conditional independence of responses.

Computational complexity per candidate item is $O(2^K)$.

Value

Numeric scalar representing the PWKL information value.

References

Cheng, Y. (2009). When cognitive diagnosis meets computerized adaptive testing: CD-CAT. *Psychometrika*, 74(4), 619–632. doi:10.1007/s1133600991232

select_next_item

Select the next item in a CD-CAT session

Description

Computes the item selection criterion score for all available items and returns the index of the best candidate.

Usage

```
select_next_item(
  items,
  responses,
  administered,
  criterion,
  est = NULL,
```

```

method = "MAP",
prior = NULL,
content = NULL,
content_prop = NULL,
exposure = NULL,
constr_fun = NULL,
initial_profile = NULL,
return_details = FALSE
)

```

Arguments

items	A cdcats_items object.
responses	Numeric vector of 0/1/NA responses (length = n_items).
administered	Integer vector of already administered item indices.
criterion	A string: "KL", "PWKL", "MPWKL", "SHE", "SEQ", or "RANDOM".
est	A list returned by <code>estimate_alpha()</code> . If NULL, computed internally.
method	Estimation method passed to <code>estimate_alpha()</code> if est = NULL.
prior	Prior probabilities passed to <code>estimate_alpha()</code> if est = NULL.
content	Character vector of length J with content domain labels. NULL disables content balancing.
content_prop	Named numeric vector of target domain proportions. NULL disables content balancing.
exposure	Numeric vector of length J with exposure parameters. NULL disables exposure control.
constr_fun	Constraint function for shadow mode, or NULL.
initial_profile	Binary integer vector of length K for first-item anchor override, or NULL.
return_details	Logical. If TRUE, returns a named list with selection details instead of a plain integer. Default FALSE.

Details

Greedy mode (constr_fun = NULL, default):

1. Filter candidates by content balancing;
2. Compute adaptive criterion scores for filtered candidates;
3. Select via exposure control.

Shadow mode (constr_fun provided):

1. Compute criterion scores for all J items (full bank);
2. Delegate selection to `constr_fun(scores, items, administered)`.

Value

When `return_details = FALSE` (default): integer scalar – item index.

When `return_details = TRUE`: named list with fields:

`item` Integer. Selected item index.

`candidate_items` Integer vector after content filter.

`criterion_scores` Named numeric vector of scores (names are item indices), or NULL for non-adaptive criteria.

`item_pre_exposure` Integer. Item that would have been selected without exposure control, or NA if not applicable.

`exposure_redirected` Logical. Whether exposure control changed the selected item.

References

Kingsbury, G. G., & Zara, A. R. (1991). A comparison of procedures for content-sensitive item selection in computerized adaptive testing. *Applied Measurement in Education*, 4(3), 241–261. doi:10.1207/s15324818ame0403_4

van der Linden, W. J., & Veldkamp, B. P. (2004). Constraining item exposure in computerized adaptive testing with shadow tests. *Journal of Educational and Behavioral Statistics*, 29(3), 273–291. doi:10.3102/10769986029003273

van der Linden, W. J. (2022). Review of the shadow-test approach to adaptive testing. *Behaviormetrika*, 49(2), 169–190. doi:10.1007/s4123702100150y

See Also

[apply_content_balancing\(\)](#), [apply_exposure_control\(\)](#), [CdcatSession](#)

SHE_criteria

Shannon Entropy (SHE) Criterion for CD-CAT

Description

Computes the expected posterior Shannon entropy after administering a candidate item.

Usage

```
SHE_criteria(item_index, prob_matrix, posterior)
```

Arguments

`item_index` Integer. Index of the candidate item.

`prob_matrix` Numeric matrix of dimension $J \times 2^K$ containing item response probabilities.

`posterior` Numeric vector of posterior probabilities.

Details

The selected item minimizes the expected posterior entropy after observing the potential response to the candidate item.

Let $\pi_t(\alpha_c)$ denote the posterior distribution over latent profiles after t administered items.

Shannon entropy is defined as:

$$SH(\pi_t) = - \sum_{c=1}^{2^K} \pi_t(\alpha_c) \log \pi_t(\alpha_c)$$

For candidate item h , the expected posterior entropy is:

$$E[SH(\pi_{t+1})] = \sum_{x=0}^1 SH(\pi_{t+1} | X_h = x) P(X_h = x | x^{(t)})$$

Thus, the criterion evaluates the expected reduction in uncertainty about the latent profile induced by administering item h .

Computational complexity per candidate item is $O(2^K)$.

Value

Numeric scalar representing the expected posterior entropy.

References

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(3), 379–423. doi:10.1002/j.15387305.1948.tb01338.x

Index

* datasets

- cdcat_sim, 13

- apply_content_balancing, 2
- apply_content_balancing(), 23
- apply_exposure_control, 4
- apply_exposure_control(), 5, 6, 23
- apply_randomesque, 5
- apply_randomesque(), 5
- apply_sympson_hetter, 6
- apply_sympson_hetter(), 5

- build_skill_patterns, 6
- build_skill_patterns(), 12

- cdcat_items, 11
- cdcat_prior, 12
- cdcat_prior(), 9
- cdcat_sim, 13
- CdcatSession, 3, 5, 7, 12, 23
- check_stopping, 15

- estimate_alpha, 17
- estimate_alpha(), 12, 22

- get_prob_matrix, 18

- KL_criteria, 18

- MPWKL_criteria, 19

- PWKL_criteria, 20

- select_next_item, 21
- select_next_item(), 3, 5
- SHE_criteria, 23