Package 'ConverTier2'

November 16, 2026

Type Package

Title Conversion of Tiered Assessment Responses to Polytomous Scores

Version 0.1.1

Description Provides tools for converting binary responses from tiered assessment instruments into polytomous item scores based on user-defined scoring rules. The package supports scoring rule parsing, data validation, tiered response conversion, and Excel file processing. It also provides a bilingual Indonesian and English graphical user interface using 'shiny', along with downloadable 'Excel' templates and example datasets.

License MIT + file LICENSE

Encoding UTF-8

Imports readxl, shiny, stats, writexl

Suggests DT, openxlsx, shinyjs, shinythemes, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Bhilan Rhiyu Antama [aut, cre],
Heni Setiyaningsih [aut],
Edi Istiyono [aut]

Maintainer Bhilan Rhiyu Antama <bhilanrhiyuantama@gmail.com>

Repository CRAN

Date/Publication 2026-11-16 09:30:15 UTC

Contents

convert_file 2
convert_tier 2
create_example_data 3
create_template_data 3
launch_app 4
parse_rules 4
validate_data 5

Index **6**

`convert_file` *Convert Tiered Assessment Excel File*

Description

Reads an Excel file containing binary tier responses, validates the data, converts the responses into item scores, and optionally writes the result to a new Excel file.

Usage

```
convert_file(input, output = NULL, rules, n_tier)
```

Arguments

`input`	Path to the input Excel file.
`output`	Path to the output Excel file. If NULL, the result is returned without writing a file.
`rules`	A named numeric vector containing scoring rules.
`n_tier`	Number of tiers in each item.

Value

A data frame containing the converted item scores.

`convert_tier` *Convert Tiered Responses to Scores*

Description

Converts binary responses from a tiered assessment into item scores using user-defined scoring rules.

Usage

```
convert_tier(data, rules, n_tier)
```

Arguments

`data`	A data frame containing binary tier responses.
`rules`	A named numeric vector containing scoring rules.
`n_tier`	Number of tiers in each item.

Value

A data frame containing item scores.

create_example_data *Create Example Tiered-Response Data*

Description

Creates reproducible example data containing binary responses (0 and 1). The default data contain 10 respondents and 10 items.

Usage

```
create_example_data(n_items = 10, n_tier = 2, n_respondents = 10)
```

Arguments

n_items	Number of items. Defaults to 10.
n_tier	Number of response tiers per item. Must be a positive integer.
n_respondents	Number of respondents. Defaults to 10.

Value

A data frame containing binary responses arranged by item and tier.

create_template_data *Create an Empty Tiered-Response Data Template*

Description

Creates an empty data frame with columns arranged by item and tier. The default template contains 10 items.

Usage

```
create_template_data(n_items = 10, n_tier = 2)
```

Arguments

n_items	Number of items. Defaults to 10.
n_tier	Number of response tiers per item. Must be a positive integer.

Value

An empty data frame with one column for each item-tier combination.

launch_app *Launch ConverTier2 Shiny Application*

Description

Launches the graphical user interface for ConverTier2.

Usage

```
launch_app()
```

Value

This function launches the Shiny application and does not return a value.

parse_rules *Parse Tier Scoring Rules*

Description

Converts text-based scoring rules into a named numeric vector.

Usage

```
parse_rules(text)
```

Arguments

text	Character string containing one scoring rule per line.

Value

A named numeric vector containing scoring rules.

Examples

```
rules <- parse_rules("
11 = 4
10 = 3
01 = 2
00 = 1
")

rules
```

validate_data *Validate Tiered Assessment Data*

Description

Checks whether a data frame is suitable for tiered binary response conversion.

Usage

```
validate_data(data, n_tier)
```

Arguments

data	A data frame containing binary tier responses.
n_tier	Number of tiers in each item.

Value

Invisibly returns TRUE if the data are valid.

Examples

```
data <- data.frame(
  Tier1_Item1 = c(1, 1, 0),
  Tier2_Item1 = c(1, 0, 0),
  Tier1_Item2 = c(0, 1, 1),
  Tier2_Item2 = c(1, 0, 1)
)

validate_data(data, n_tier = 2)
```

Index

convert_file, 2
convert_tier, 2
create_example_data, 3
create_template_data, 3

launch_app, 4

parse_rules, 4

validate_data, 5