Package ‘airt’

August 8, 2023

Type Package
Title Evaluation of Algorithm Collections Using Item Response Theory
Version 0.2.2
Maintainer Sevvandi Kandanaarachchi <sevvandik@gmail.com>
Description An evaluation framework for algorithm portfolios using Item Response Theory (IRT). We use continuous and polytomous IRT models to evaluate algorithms and introduce algorithm characteristics such as stability, effectiveness and anomalousness (Kandanaarachchi, Smith-Miles 2020) <doi:10.13140/RG.2.2.11363.09760>.
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.2.1
Depends R (>= 3.4.0)
Imports pracma, mirt, tidyr, EstCRM, rlang, dplyr, magrittr, tibble, ggplot2, grDevices, RColorBrewer
Suggests knitr, rmarkdown, gridExtra, scales
VignetteBuilder knitr
URL https://sevvandi.github.io/airt/
NeedsCompilation no
Author Sevvandi Kandanaarachchi [aut, cre] (<https://orcid.org/0000-0002-0337-0395>)
Repository CRAN
Date/Publication 2023-08-08 12:50:09 UTC

R topics documented:

algo_effectiveness_crm ........................................... 2
algo_effectiveness_poly ....................................... 3
cirtmodel .......................................................... 4
algo_effectiveness_crm

Computes the actual and predicted effectiveness of a given algorithm.

Description

This function computes the actual and predicted effectiveness of a given algorithm for different tolerance values.

Usage

algo_effectiveness_crm(mod, num = 1)

Arguments

mod A fitted mirt model using the function irtmodel or R package mirt.
num The algorithm number, for which the goodness of the IRT model is computed.

Value

A list with the following components:

effective The x,y coordinates for the actual and predicted effectiveness curves for algorithm num.
predictedEff The area under the predicted effectiveness curve.
actualEff The area under the actual effectiveness curve.
Examples

```r
set.seed(1)
x1 <- runif(100)
x2 <- runif(100)
x3 <- runif(100)
X <- cbind.data.frame(x1, x2, x3)
max_item <- rep(1,3)
min_item <- rep(0,3)
mod <- cirtmodel(X, max.item=max_item, min.item=min_item)
out <- algo_effectiveness_crm(mod$model, num=1)
out
```

algo_effectiveness_poly

*Computes the actual and predicted effectiveness of a given algorithm.*

Description

This function computes the actual and predicted effectiveness of a given algorithm for different tolerance values.

Usage

```r
algo_effectiveness_poly(mod, num = 1)
```

Arguments

- **mod**: A fitted mirt model using the function irtmodel or R package mirt.
- **num**: The algorithm number

Value

A list with the following components:

- `effective`: The x,y coordinates for the actual and predicted effectiveness curves for algorithm num.
- `predictedEff`: The area under the predicted effectiveness curve.
- `actualEff`: The area under the actual effectiveness curve.

```r
#' @examples set.seed(1) x1 <- sample(1:5, 100, replace = TRUE) x2 <- sample(1:5, 100, replace = TRUE) x3 <- sample(1:5, 100, replace = TRUE) X <- cbind.data.frame(x1, x2, x3) mod <- cirtmodel(X) out <- algo_effectiveness_poly(mod$model, num=1) out
```
cirtmodel  

Fits a continuous IRT model.

Description
This function fits a continuous Item Response Theory (IRT) model to the algorithm performance data. The function EstCRMitem in the R package EstCRM is updated to accommodate negative discrimination.

Usage

\[
cirtmodel(df, max.item = NULL, min.item = NULL)
\]

Arguments

df  
The performance data in a matrix or dataframe.
max.item  
A vector with the maximum performance value for each algorithm.
min.item  
A vector with the minimum performance value for each algorithm.

Value

A list with the following components:

model  
The IRT model.
anomalous  
A binary value for each algorithm. It is set to 1 if an algorithm is anomalous. Otherwise it is set to 0.
consistency  
The consistency of each algorithm.
difficulty_limit  
The difficulty limit of each algorithm. A higher difficulty limit indicates that the algorithm can tackle harder problems.

References


Examples

\[
\text{set.seed(1)}
\text{x1 <- runif(100)}
\text{x2 <- runif(100)}
\text{x3 <- runif(100)}
\text{X <- cbind.data.frame(x1, x2, x3)}
\text{mod <- cirtmodel(X)}
\]
**classification_cts**

A dataset containing classification algorithm performance data in a continuous format.

**Description**

This dataset contains the performance of 10 classification algorithms on 235 datasets discussed in the paper Instance Spaces for Machine Learning Classification by M. A. Munoz, L. Villanova, D. Baatar, and K. A. Smith-Miles.

**Usage**

classification_cts

**Format**

A dataframe of 235 x 10 dimensions.

**Dimension 1** Each row contains the algorithm performance of a dataset on 10 classification algorithms.

**Dimensions 2** Each column contains the algorithm performance of a single algorithm.

**Source**

https://katesmithmiles.wixsite.com/home/matilda

---

**classification_poly**

A dataset containing classification algorithm performance data in a polytomous format.

**Description**

This dataset contains the performance of 10 classification algorithms on 235 datasets discussed in the paper Instance Spaces for Machine Learning Classification by M. A. Munoz, L. Villanova, D. Baatar, and K. A. Smith-Miles.

**Usage**

classification_poly

**Format**

A dataframe of 235 x 10 dimensions.

**Dimension 1** Each row contains the algorithm performance of a dataset on 10 classification algorithms.

**Dimensions 2** Each column contains the algorithm performance of a single algorithm.
**effectiveness_crm**

Computes the actual and predicted effectiveness of the collection of algorithms.

**Description**

This function computes the actual and predicted effectiveness of the collection of algorithms for different tolerance values.

**Usage**

```r
effectiveness_crm(model)
```

## S3 method for class 'effectiveness_crm'

```r
autoplot(object, plottype = 1, ...)
```

**Arguments**

- `model` The output of the function cirtmodel.
- `object` For autoplot: The output of the function effectiveness_crm
- `plottype` For autoplot: If plottype = 1, then actual effectiveness is plotted, if plottype = 2, then predicted effectiveness is plotted. If plottype = 3, area under the actual effectiveness curve (AUAEV) is plotted against area under the predicted effectiveness curve (AUPEC).
- `...` Other arguments currently ignored.

**Value**

A list with the following components:

- `effectivenessAUC` The area under the actual and predicted effectiveness curves.
- `actcurves` The x,y coordinates for the actual effectiveness curves for each algorithm.
- `prdcurves` The x,y coordinates for the predicted effectiveness curves for each algorithm.
**effectiveness_poly**  

*Computes the actual and predicted effectiveness of the collection of algorithms.*

**Description**

This function computes the actual and predicted effectiveness of the collection of algorithms for different tolerance values.

**Usage**

```r
effectiveness_poly(model)
```

```r
## S3 method for class 'effectivenesspoly'
autoplot(object, plottype = 1, ...)
```

**Arguments**

- `model`  
  The output of pirtmodel function.

- `object`  
  For autoplot: The output of the function effectiveness_crm

- `plottype`  
  For autoplot: If plottype = 1, then actual effectiveness is plotted, if plottype = 2, then predicted effectiveness is plotted. If plottype = 3, area under the actual effectiveness curve (AUAEC) is plotted against area under the predicted effectiveness curve (AUPEC).

- `...`  
  Other arguments currently ignored.
Value

A list with the following components:

- **effectivenessAUC**: The area under the actual and predicted effectiveness curves.

- **actcurves**: The x,y coordinates for the actual effectiveness curves for each algorithm.

- **prdcurves**: The x,y coordinates for the predicted effectiveness curves for each algorithm.

Examples

```r
set.seed(1)
x1 <- sample(1:5, 100, replace = TRUE)
x2 <- sample(1:5, 100, replace = TRUE)
x3 <- sample(1:5, 100, replace = TRUE)
X <- cbind.data.frame(x1, x2, x3)
mod <- pirtmodel(X)
out <- effectiveness_poly(mod)
out
# For actual effectiveness curves
autoplot(out, plottype = 1)
# For predicted effectiveness curves
autoplot(out, plottype = 2)
# For Actual and Predicted Effectiveness (AUAEC, AUPEC)
autoplot(out, plottype = 3)
```

### heatmaps_crm

*Function to produce heatmaps from a continuous IRTmodel*

**Description**

This function makes a dataframe from the continuous IRTmodel the autoplot function produces the heatmaps.

**Usage**

```r
heatmaps_crm(model, thetarange = c(-6, 6))
```

# S3 method for class 'heatmapcrm'

```r
autoplot(
    object,
    xlab = "Theta",
    nrow = 2,
    ratio = 1,
    col_scheme = "plasma",
    ...
)
```
**latent_trait_analysis**

**Arguments**

- `model`: Output from the function `cirtmodel`.
- `thetarange`: The range for theta, default from -6 to 6.
- `object`: For autoplot: output of `heatmaps_crm` function.
- `xlab`: For autoplot: xlabel.
- `nrow`: For autoplot: number of rows of heatmaps to plot.
- `ratio`: For autoplot: ratio for `coord_fixed` in `ggplot`.
- `col_scheme`: For autoplot: the color scheme for heatmaps. Default value is plasma.
- `...`: Other arguments currently ignored.

**Value**

Dataframe with output probabilities from the IRT model for all algorithms, an object of class `heatmapcrm`.

**Examples**

```r
data(classification_cts)
model <- cirtmodel(classification_cts)
obj <- heatmaps_crm(model)
head(obj$df)
autoplot(obj)
```

---

**latent_trait_analysis**  Performs the latent trait analysis

**Description**

This function performs the latent trait analysis of the datasets/problems after fitting a continuous IRT model. It fits a smoothing spline to the points to compute the latent trait. The autoplot function plots the latent trait and the performance.

**Usage**

```r
latent_trait_analysis(
  df,  
  paras, 
  min_item = NULL, 
  max_item = NULL, 
  epsilon = 0.01
)
```

```r
## S3 method for class 'latenttrait'
autoplot(
```

---

```r
```
object, xlab = "Problem Difficulty", ylab = "Performance", plottype = 1, nrow = 2, se = TRUE, ratio = 3, ... 
)

Arguments

df The performance data in a matrix or dataframe.
paras The parameters from fitting cirtmodel.
min_item A vector with the minimum performance value for each algorithm.
max_item A vector with the maximum performance value for each algorithm.
epsilon A value defining good algorithm performance. If epsilon = 0, then only the best algorithm is considered. A default
object For autoplot: the output of the function latent_trait_analysis.

xlab For autoplot: the xlabel.
ylab For autoplot: the ylabel.

plottype For autoplot: plottype = 1 for all algorithm performances in a single plot, plottype = 2 for using facet_wrap to plot individual algorithms, plottype = 3 to plot the smoothing splines and plottype = 4 to plot strengths and weaknesses.
nrow For autoplot: If plottype = 2, the number of rows for facet_wrap.
se For autoplot: for plotting splines with standard errors.
ratio For autoplot: for plotting strengths and weaknesses, ratio between x and y axis.
... Other arguments currently ignored.

Value

A list with the following components:

crmtheta The problem trait output computed from the R package EstCRM.

strengths The strengths of each algorithm and positions on the latent trait that they perform well.

longdf The dataset in long format of latent trait occupancy.

plt The ggplot object showing the fitted smoothing splines.

widedf The dataset in wide format with latent trait.

thetas The easiness of the problem set instances.

weakness The weaknesses of each algorithm and positions on the latent trait that they perform poorly.
Examples

```r
# This is a dummy example.
set.seed(1)
x1 <- runif(200)
x2 <- 2*x1 + rnorm(200, mean=0, sd=0.1)
x3 <- 1 - x1 + rnorm(200, mean=0, sd=0.1)
X <- cbind.data.frame(x1, x2, x3)
max_item <- rep(max(x1, x2, x3),3)
min_item <- rep(min(x1, x2, x3),3)
mod <- cirtmodel(X, max.item=max_item, min.item=min_item)
out <- latent_trait_analysis(X, mod$model$param, min_item= min_item, max_item = max_item)
out
# To plot performance against the problem difficulty
autoplot(out)
# To plot individual panels
autoplot(out, plottype = 2)
# To plot smoothing splines
autoplot(out, plottype = 3)
# To plot strengths and weaknesses
autoplot(out, plottype = 4)
```

make_polyIRT_data

Converts continuous performance data to polytomous data with 5 categories.

Description

This function converts continuous performance data to polytomous data with 5 categories.

Usage

`make_polyIRT_data(df, method = 1)`

Arguments

- `df`: The input data in a dataframe or a matrix
- `method`: If 1, then the data is an accuracy measure between 0 and 1. If 2, then the performance data is possibly has a bigger range. So we divide it into 5 equal bins to make it polytomous.

Value

The polytomous data frame.
Examples

```r
set.seed(1)
x1 <- runif(500)
x2 <- runif(500)
x3 <- runif(500)
x <- cbind(x1, x2, x3)
xout <- make_polyIRT_data(x)
```

---

```r
model_goodness_crm(model)
```

Computes the goodness of IRT model for all algorithms.

---

Description

This function computes the goodness of the IRT model for all algorithms for different goodness tolerances.

Usage

```r
model_goodness_crm(model)
```

## S3 method for class 'modelgoodnesscrm'
```r
autoplot(object, ...)
```

Arguments

- `model`: The output of function `cirtmodel`.
- `object`: For `autoplot`: The output of `model_goodness_crm`.
- `...`: Other arguments currently ignored.

Value

A list with the following components:

- `goodnessAUC`: The area under the model goodness curve for each algorithm.
- `curves`: The x, y coordinates for the model goodness curves for each algorithm.
- `residuals`: The residuals for each algorithm using the AIRT model.

Examples

```r
set.seed(1)
x1 <- runif(200)
x2 <- 2*x1 + rnorm(200, mean=0, sd=0.1)
x3 <- 1 - x1 + rnorm(200, mean=0, sd=0.1)
X <- cbind.data.frame(x1, x2, x3)
mod <- cirtmodel(X)
```
model_goodness_for_algo_crm

out <- model_goodness_crm(mod)
out
autoplot(out)

model_goodness_for_algo_crm

Computes the goodness of IRT model for a given algorithm.

Description

This function computes the goodness of the IRT model for a given algorithm for different goodness tolerances.

Usage

model_goodness_for_algo_crm(mod, num = 1)

Arguments

mod A fitted mirt model using the function irtmodel or R package mirt.
num The algorithm number, for which the goodness of the IRT model is computed.

Value

A list with the following components:

xy The x values denote the goodness tolerances. The y values denote the model goodness.
auc The area under the model goodness curve.
residuals The different between actual and fitted performance values.

Examples

set.seed(1)
x1 <- runif(100)
x2 <- runif(100)
x3 <- runif(100)
X <- cbind.data.frame(x1, x2, x3)
max_item <- rep(1,3)
min_item <- rep(0,3)
mod <- cirtmodel(X, max.item=max_item, min.item=min_item)
out <- model_goodness_for_algo_crm(mod$model, num=1)
out
model_goodness_for_algo_poly

*Computes the goodness of the IRT model fit for a given algorithm.*

**Description**

This function computes the goodness of the IRT model fit for a given algorithm using the empirical cumulative distribution function of errors.

**Usage**

```r
model_goodness_for_algo_poly(mod, num = 1)
```

**Arguments**

- `mod` A fitted mirt model using the function irtmodel or R package mirt.
- `num` The algorithm number

**Value**

A list with the following components:

- `xy` The x values denote the error tolerances. The y values denotes its empirical cumulative distribution function.
- `auc` The area under the CDF.
- `mse` The mean squared error.

**Examples**

```r
set.seed(1)
x1 <- sample(1:5, 100, replace = TRUE)
x2 <- sample(1:5, 100, replace = TRUE)
x3 <- sample(1:5, 100, replace = TRUE)
X <- cbind.data.frame(x1, x2, x3)
mod <- pirtmodel(X)
out <- model_goodness_for_algo_poly(mod$model, num = 1)
out
```
model_goodness_poly  Computes the goodness of IRT model for all algorithms.

Description

This function computes the goodness of the IRT model for all algorithms using the empirical cumulative distribution function of errors.

Usage

model_goodness_poly(model)

## S3 method for class 'modelgoodnesspoly'
autoplot(object, ...)

Arguments

model  The output from pirtmodel function.
object  For autoplot: The output of the model_goodness_poly function.
...  Other arguments currently ignored.

Value

A list with the following components:

goodnessAUC  The area under the model goodness curve for each algorithm.
mse  The mean squared error.
curves  The x,y coordinates for the model goodness curves for each algorithm.

Examples

set.seed(1)
x1 <- sample(1:5, 100, replace = TRUE)
x2 <- sample(1:5, 100, replace = TRUE)
x3 <- sample(1:5, 100, replace = TRUE)
X <- cbind.data.frame(x1, x2, x3)
mod <- pirtmodel(X)
out <- model_goodness_poly(mod)
out
autoplot(out)
pirtmodel

*Fits a polytomous IRT model.*

**Description**

This function fits a polytomous Item Response Theory (IRT) model using the R package mirt to the algorithm performance data.

**Usage**

```r
pirtmodel(dat, ncycle = NULL, vpara = TRUE)
```

**Arguments**

- `dat` The performance data in a matrix or dataframe.
- `ncycle` The number of cycles for mirt. The default is 500.
- `vpara` If TRUE the verbose parameter for the mirt would be set to true.

**Value**

A list with the following components:

- `model` The IRT model using the R package mirt.
- `anomalous` A binary value for each algorithm. It is set to 1 if an algorithm is anomalous. Otherwise it is set to 0.
- `consistency` The consistency of each algorithm.
- `difficulty_limit` The difficulty limits for each algorithm. A higher threshold indicates that the algorithm can tackle harder problems.

**References**


**Examples**

```r
set.seed(1)
x1 <- sample(1:5, 100, replace = TRUE)
x2 <- sample(1:5, 100, replace = TRUE)
x3 <- sample(1:5, 100, replace = TRUE)
X <- cbind.data.frame(x1, x2, x3)
mod <- pirtmodel(X)
```
tracelines_poly  

Function to plot tracelines from a polytomous IRT model

Description

This function makes a dataframe from the polytomous IRT model. The autoplot function can be used to plot trace lines

Usage

tracelines_poly(model)

## S3 method for class 'tracelinespoly'

autoplot(
    object,
    xlab = "Theta",
    ylab = "Probability",
    nrow = 2,
    title = "Tracelines",
    ...
)

Arguments

model  
Output from the function pirtmodel.

object  
For autoplot: output of tracelines_poly function.

xlab  
For autoplot: xlabel.

ylab  
For autoplot: ylabel.

nrow  
For autoplot: number of rows of heatmaps to plot.

title  
For autoplot: the title for the plot.

...  
Other arguments currently ignored.

Value

Dataframe with output probabilities from the IRT model for all algorithms, an object of the class tracelinespoly.

Examples

data(classification_poly)
mod <- pirtmodel(classification_poly)
obj <- tracelines_poly(mod)
head(obj$df)
autoplot(obj)
Index

* datasets
  classification_cts, 5
  classification_poly, 5

algo_effectiveness.crm, 2
algo_effectiveness.poly, 3
 autoplot.effectivenesscrm
    (effectiveness.crm), 6
 autoplot.effectivenesspoly
    (effectiveness.poly), 7
 autoplot.heatmapcrm (heatmaps.crm), 8
 autoplot.latenttrait
    (latent_trait_analysis), 9
 autoplot.modelgoodnesscrm
    (model_goodness.crm), 12
 autoplot.modelgoodnesspoly
    (model_goodness.poly), 15
 autoplot.tracelinespoly
    (tracelines_poly), 17

cirtmodel, 4
classification_cts, 5
classification_poly, 5

effectiveness.crm, 6
effectiveness.poly, 7

heatmaps.crm, 8

latent_trait_analysis, 9

make_polyIRT_data, 11
model_goodness.crm, 12
model_goodness_for.algo.crm, 13
model_goodness_for.algo.poly, 14
model_goodness_poly, 15

pirtmodel, 16

tracelines_poly, 17

18