Package ‘bamdit’

September 17, 2018

Type Package
Title Bayesian Meta-Analysis of Diagnostic Test Data
Version 3.2.1
Date 2018-09-14
Depends R (>= 3.4.0)
Imports rjags (>= 3.4), R2jags (>= 0.04-03), ggplot2 (>= 3.0.0),
ggExtra (>= 0.8), MASS (>= 7.3), grid, gridExtra (>= 2.3)
SystemRequirements JAGS (>= 3.4.0) (see http://mcmc-jags.sourceforge.net)
Description Functions for Bayesian meta-analysis of diagnostic test data which are based on a scale mixtures bivariate random-effects model.
License GPL (>= 2)
Repository CRAN
RoxygenNote 6.1.0
Encoding UTF-8
NeedsCompilation no
Author Pablo Emilio Verde [aut, cre]
Maintainer Pablo Emilio Verde <pabloemilio.verde@hhu.de>
Date/Publication 2018-09-17 15:00:03 UTC

R topics documented:

  bamdit-package .................................................. 2
  bsroc .......................................................... 3
  ct ........................................................... 4
  ep ........................................................... 5
  glas ........................................................... 6
  gould ......................................................... 7
  metadiag ...................................................... 7
  mri ........................................................... 14
  plot.metadiag .................................................. 15
Bayesian meta-analysis of diagnostic test data based on a scale mixtures bivariate random-effects model. This package was developed with the aim of simplifying the use of meta-analysis models that up to now have demanded great statistical expertise in Bayesian meta-analysis. The package implements a series of innovative statistical techniques including: the BSROC (Bayesian Summary ROC) curve, the BAUC (Bayesian AUC), predictive surfaces, the use of prior distributions that avoid boundary estimation problems of component of variance and correlation parameters, analysis of conflict of evidence and robust estimation of model parameters. In addition, the package comes with several published examples of meta-analysis that can be used for illustration or further research in this area.

Details

Package: bamdit
Type: Package
Version: 3.2.1
Date: 2018-09-14
License: GPL (>= 2)
LazyLoad: yes

Author(s)

PD Dr. Pablo Emilio Verde <pabloemilio.verde@hhu.de>

References

Description

This function plots the observed data in the ROC (Receiving Operating Characteristics) space with the Bayesian SROC (Summary ROC) curve. The predictive curves are approximated using a parametric model.

Usage

```r
bsroc(m, level = c(0.05, 0.5, 0.95), title = "Bayesian SROC Curve", fpr.x = seq(0.01, 0.95, 0.01), partial.AUC = TRUE, xlim.bsroc = c(0, 1), ylim.bsroc = c(0, 1), lower.auc = 0, upper.auc = 0.95, col.fill.points = "blue", results.bauc = TRUE, results.bsroc = FALSE, plot.post.bauc = FALSE, bins = 30, scale.size.area = 10)
```

Arguments

- **m**: The object generated by metadiag.
- **level**: Credibility levels of the predictive curve.
- **title**: Optional parameter for setting a title in the plot.
- **fpr.x**: Grid of values where the conditionl distribution is calculated.
- **partial.AUC**: Automatically calculate the AUC for the observed range of FPRs, default is TRUE.
- **xlim.bsroc**, **ylim.bsroc**: Graphical limits of the x-axis and y-axis for the BSROC curve plot.
- **lower.auc**, **upper.auc**: Lower and upper limit of the AUC.
- **col.fill.points**: Color used to fill points, default is blue.
- **results.bauc**: Print results of the Bayesian Area Under the Curve, default value is TRUE.
- **results.bsroc**: Print results of the Bayesian SROC curve, default value is FALSE.
- **plot.post.bauc**: The BSROC and the posterior of the BAUC are plotted in the same page, default is FALSE.
- **bins**, **scale.size.area**: Histograms’ bins. Scale area for the plotted points, default = 10.
ct

Diagnosis of appendicities with computer tomography scans

Description

This data frame corresponds to 51 clinical studies reporting the accuracy of computer tomography (CT) scans for the diagnosis of appendicities.

Format

A matrix with 51 rows and 16 columns. Each row represents study results, the columns are:

- \textbf{tp} number of true positives.
- \textbf{n1} number of patients with disease.
- \textbf{fp} number of false positives.

References


See Also

\texttt{metadiag}.

Examples

```r
## execute analysis
## Not run:
# Example: data from Glas et al. (2003)

data(glas)
glas.t <- glas[glas$marker == "Telomerase", 1:4]
glas.m1 <- metadiag(glas.t, re = "normal", link = "logit")
bsroc(glas.m1)
bsroc(glas.m1, plot.post.bauc = TRUE)

# Example: data from Scheidler et al. (1997)
# In this example the range of the observed FPR is less than 20%.
# Calculating the BSROC curve makes no sense! You will get a warning message!

data(mri)
mri.m <- metadiag(mri)
bsroc(mri.m)

## End(Not run)
```
**Details**

This data frame corresponds to 51 clinical studies reporting the accuracy of computer tomography (CT) scans for the diagnosis of appendicitis.

**Source**

The data were obtained from


**References**


---

**Ectopic pregnancy vs. all other pregnancies data**

**Description**

Ectopic pregnancy vs. all other pregnancies data Table III Mol et al. 1998
Format

A matrix with 21 rows and 8 columns. Each row represents study results, the columns are:

- **tp**: number of true positives.
- **n1**: number of patients with disease.
- **fp**: number of false positives.
- **n2**: number of patients without disease.
- **d1**: Prospective vs. retrospective.
- **d2**: Cohort vs. case-control
- **d3**: Consecutive sampling patients series vs. non-consecutive.

Source

Table III Mol et al. 1998

Description

Outcome of individual studies evaluating urine markers

Format

A matrix with 46 rows and 7 columns. Each row represents study results, the columns are:

- **tp**: number of true positives.
- **n1**: number of patients with disease.
- **fp**: number of false positives.
- **n2**: number of patients without disease.
- **author**: first author of the study.
- **cutoff**: cutoff in U/ml.
- **marker**: test method used in the study.

Source

The data were obtained from


References

### Description

Data from a Meta-Analysis of Studies Quality of FDG-PET for Diagnosis of SPNs and Mass Lesions

### Format

A matrix with 31 rows and 6 columns. Each row represents study results, the columns are:

- `tp` number of true positives.
- `n1` number of patients with disease.
- `fp` number of false positives.
- `n2` number of patients without disease.
- `author` first author of the study.
- `year` publication date.

### Source

The data were obtained from


### Description

This function performs a Bayesian meta-analysis of diagnostic test data by fitting a bivariate random effects model. The number of true positives and false positives are modeled with two conditional Binomial distributions and the random-effects are based on a bivariate scale mixture of Normals. Computations are done by calling JAGS (Just Another Gibbs Sampler) to perform MCMC (Markov Chain Monte Carlo) sampling and returning an object of the class `mcmc.list`. 
metadiag

Usage

metadiag(data, two.by.two = FALSE, re = "normal", re.model = "DS",
link = "logit", mean.mu.D = 0, mean.mu.S = 0, sd.mu.D = 1,
sd.mu.S = 1, sigma.D.upper = 10, sigma.S.upper = 10,
mean.Fisher.rho = 0, sd.Fisher.rho = 1/sqrt(2), df = 4,
df.estmate = FALSE, df.lower = 3, df.upper = 20,
split.w = FALSE, n.1.new = 50, n.2.new = 50, nr.chains = 2,
nr.iterations = 10000, nr.adapt = 1000, nr.burnin = 1000,
nr.thin = 1, be.quiet = FALSE, r2jags = TRUE)

Arguments

data
  Either a data frame with at least 4 columns containing the true positives (tp),
  number of patients with disease (n1), false positives (fp), number of patients
  without disease (n2), or for two.by.two = TRUE a data frame where each line
  contains the diagnostic results as a two by two table, where the column names
  are: TP, FP, TN, FN.
two.by.two
  If TRUE indicates that the diagnostic results are given as: TP, FP, TN, FN.
re
  Random effects distribution for the resulting model. Possible values are normal
  for bivariate random effects and sm for scale mixtures.
re.model
  If re.model = "DS" indicates that the sum and differences of TPR and FPR are
  modeled as random effects and re.model = "SeSp" indicates that the Sensitivity
  and Specificity are modeled as ranodm effects. The defualt value is re.model =
  "DS".
link
  The link function used in the model. Possible values are logit, cloglog probit.
mean.mu.D
  prior Mean of D, default value is 0.
mean.mu.S
  prior Mean of S, default value is 0.
sd.mu.D
  prior Standard deviation of D, default value is 1 (the prior of mu.D is a logistic
  distribution).
sd.mu.S
  prior Standard deviation of S, default value is 1 (the prior of mu.S is a logistic
  distribution).
sigma.D.upper
  Upper bound of the uniform prior of sigma.S, default value is 10.
sigma.S.upper
  Upper bound of the uniform prior of sigma.S, default value is 10.
mean.Fisher.rho
  Mean of rho in the Fisher scale default value is 0.
sd.Fisher.rho
  Standard deviation of rho in the Fisher scale, default value is 1/sqrt(2).
df
  If de.estmate = FALSE, then df is the degrees of freedom for the scale mixture
distribution, default value is 4.
df.estmate
  Estimate the posterior of df. The default value is FALSE.
df.lower
  Lower bound of the prior of df. The defualat value is 3.
df.upper
  Upper bound of the prior of df. The defualat value is 30.
split.w
  Split the w parameter in two independent weights one for each random effect.
The default value is FALSE.
Number of patients with disease in a predictive study default is 50.

Number of patients with non-disease in a predictive study default is 50.

Number of chains for the MCMC computations, default 5.

Number of iterations after adapting the MCMC, default is 10000. Some models may need more iterations.

Number of iterations in the adaptation process, default is 1000. Some models may need more iterations during adaptation.

Number of iteration discared for burnin period, default is 1000. Some models may need a longer burnin period.

Thinning rate, it must be a positive integer, the default value 1.

Do not print warning message if the model does not adapt default value is FALSE. If you are not sure about the adaptation period choose be.quiet=TRUE.

Which interface is used to link R to JAGS (rjags and R2jags) default value is R2Jags TRUE.

Details

Installation of JAGS: It is important to note that R 3.3.0 introduced a major change in the use of toolchain for Windows. This new toolchain is incompatible with older packages written in C++. As a consequence, if the installed version of JAGS does not match the R installation, then the rjags package will spontaneously crash. Therefore, if a user works with R version >= 3.3.0, then JAGS must be installed with the installation program JAGS-4.2.0-Rtools33.exe. For users who continue using R 3.2.4 or an earlier version, the installation program for JAGS is the default installer JAGS-4.2.0.exe.

Value

This function returns an object of the class metadiag. This object contains the MCMC output of each parameter and hyper-parameter in the model, the data frame used for fitting the model, the link function, type of random effects distribution and the splitting information for conflict of evidence analysis.

The results of the object of the class metadiag can be extracted with R2jags or with rjags. In addition a summary, a print and a plot functions are implemented for this type of object.

References


Examples

```r
## Not run:
```
# Example: data from Glas et al. (2003)
library(bandit)
data("glas")
glas.t <- glas[glas$marker == "Telomerase", 1:4]

glas.t <- glas[glas$marker == "Telomerase", 1:4]

# Simple visualization ...

plotdata(glas.t, # Data frame
two.by.two = FALSE # Data is given as: (tp, n1, fp, n2)
)

glas.m1 <- metadiag(glas.t, # Data frame
two.by.two = FALSE, # Data is given as: (tp, n1, fp, n2)
re = "normal", # Random effects distribution
re.model = "DS", # Random effects on D and S
link = "logit", # Link function
sd.Fisher.rho = 1.7, # Prior standard deviation of correlation
nr.burnin = 1000, # Iterations for burnin
nr.iterations = 10000, # Total iterations
nr.chains = 2, # Number of chains
r2jags = TRUE) # Use r2jags as interface to jags

summary(glas.m1, digit=3)

plot(glas.m1, # Fitted model
level = c(0.5, 0.75, 0.95), # Credibility levels
parametric.smooth = TRUE) # Parametric curve

# Plot results: based on a non-parametric smoother of the posterior predictive rates .......

plot(glas.m1, # Fitted model
level = c(0.5, 0.75, 0.95), # Credibility levels
parametric.smooth = FALSE) # Non-parametric curve

# Using the pipe command in the package dplyr ...........................................

library(dplyr)
glas.t %>%
metadiag(re = "normal", re.model = "SeSp") %>%
plot(parametric.smooth = FALSE, color.pred.points = "red")

# Visualization of posteriors of hyper-parameters ........................................
library(ggplot2)
library(GGally)
library(R2jags)
attach(jags(glas.ml))
ggpairs(hyper.post, # Data frame
title = "Hyper-Posteriors", # title of the graph
lower = list(continuous = "density") # contour plots)

# List of different statistical models:
# 1) Different link functions: logit, cloglog and probit

# 2) Different parametrization of random effects in the link scale:
#    DS = "differences of TPR and FPR"
#    SeSp = "Sensitivity and Specificity"

# 3) Different random effects distributions:
#    "normal" or "sm = scale mixtures".

# 4) For the scale mixture random effects:
#    split.w = TRUE => "split the weights".

# 5) For the scale mixture random effects:
#    df.estimate = TRUE => "estimate the degrees of freedom".

# 6) For the scale mixture random effects:
#    df.estimate = TRUE => "estimate the degrees of freedom".

# 7) For the scale mixture random effects:
#    df = 4 => "fix the degrees of freedom to a particular value".
#    Note that df = 1 fits a Cauchy bivariate distribution to the random effects.

# logit-normal-DS
m <- metadiag(glas.t, re = "normal", re.model = "DS", link = "logit")
summary(m)
plot(m)

# cloglog-normal-DS
summary(metadiag(glas.t, re = "normal", re.model = "DS", link = "cloglog"))

# probit-normal-DS
summary(metadiag(glas.t, re = "normal", re.model = "DS", link = "probit"))
# logit-normal-SeSp
summary(metadiag(glas.t, re = "normal", re.model = "SeSp", link = "logit"))
# cloglog-normal-SeSp
summary(metadiag(glas.t, re = "normal", re.model = "SeSp", link = "cloglog"))
# probit-normal-SeSp
summary(metadiag(glas.t, re = "normal", re.model = "SeSp", link = "probit"))
# logit-sm-DS
summary(metadiag(glas.t, re = "sm", re.model = "DS", link = "logit", df = 1))
```r
# cloglog-sm-DS
summary(m<metadiag(glas.t, re = "sm", re.model = "DS", link = "cloglog", df = 1))
plot(m, parametric.smooth = FALSE)

# probit-sm-DS
summary(m<metadiag(glas.t, re = "sm", re.model = "DS", link = "probit", df = 1))
plot(m, parametric.smooth = FALSE)

# logit-sm-SeSp
summary(m<metadiag(glas.t, re = "sm", re.model = "SeSp", link = "logit", df = 1))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# cloglog-sm-SeSp
summary(m<metadiag(glas.t, re = "sm", re.model = "SeSp", link = "cloglog", df = 1))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# probit-sm-SeSp
summary(m<metadiag(glas.t, re = "sm", re.model = "SeSp", link = "probit", df = 1))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# logit-sm-DS-df
summary(m<metadiag(glas.t, re = "sm", re.model = "DS", link = "logit", df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# cloglog-sm-DS-df
summary(m<metadiag(glas.t, re = "sm", re.model = "DS", link = "cloglog", df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# probit-sm-DS-df
summary(m<metadiag(glas.t, re = "sm", re.model = "DS", link = "probit", df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# logit-sm-SeSp-df
summary(m<metadiag(glas.t, re = "sm", re.model = "SeSp", link = "probit", df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# cloglog-sm-SeSp-df
summary(m<metadiag(glas.t, re = "sm", re.model = "SeSp", link = "cloglog", df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# probit-sm-SeSp-df
summary(m<metadiag(glas.t, re = "sm", re.model = "SeSp", link = "probit", df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))

# split.w .................................................................
```
```r
# logit-sm-DS
summary(m <- metadiag(glas.t, re = "sm", re.model = "DS", link = "logit", split.w = TRUE, df = 10))
plot(m)

# cloglog-sm-DS
summary(m <- metadiag(glas.t, re = "sm", re.model = "DS", link = "cloglog", split.w = TRUE, df = 4))
plot(m)

# probit-sm-DS
summary(m <- metadiag(glas.t, re = "sm", re.model = "DS", link = "probit", split.w = TRUE, df = 4))
plot(m, parametric.smooth = FALSE)

# logit-sm-SeSp
summary(m <- metadiag(glas.t, re = "sm", re.model = "SeSp", link = "logit", split.w = TRUE, df = 1))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))
plot(m)

# cloglog-sm-SeSp
summary(m <- metadiag(glas.t, re = "sm", re.model = "SeSp", link = "cloglog", split.w = TRUE, df = 1))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))
plot(m)

# probit-sm-SeSp
summary(m <- metadiag(glas.t, re = "sm", re.model = "SeSp", link = "probit", split.w = TRUE, df = 1))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))
plot(m)

# logit-sm-DS-df
summary(m <- metadiag(glas.t, re = "sm", re.model = "DS", link = "logit", split.w = TRUE,
df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))
plot(m)

# cloglog-sm-DS-df
summary(m <- metadiag(glas.t, re = "sm", re.model = "DS", link = "cloglog", split.w = TRUE,
df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))
plot(m)

# probit-sm-DS-df
summary(m <- metadiag(glas.t, re = "sm", re.model = "DS", link = "probit", split.w = TRUE,
df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))
plot(m)

# logit-sm-SeSp-df
summary(m <- metadiag(glas.t, re = "sm", re.model = "SeSp", link = "probit", split.w = TRUE,
df.estimate = TRUE))
plot(m, parametric.smooth = FALSE, level = c(0.5, 0.9))
plot(m)

# cloglog-sm-SeSp-df
```
Summary

Diagnosis of lymph node metastasis with magnetic resonance imaging

Description

Diagnosis of lymph node metastasis with magnetic resonance imaging

Format

A matrix with 10 rows and 4 columns. Each row represents study results, the columns are:

- **tp** true positives
- **n1** number of patients with disease
- **fp** false positives
- **n2** number of patients without disease

Source


References

Generic plot function for metadiag object in bamdit

Description

This function plots the observe data in the ROC (Receiving Operating Characteristics) space with the posterior predictive contours. The predictive curves are approximated using a non-parametric smoother or with a parametric model. For the parametric model the current implementation supports only a logistic link function. The marginal posterior predictive distributions are plotted outside the ROC space.

Usage

```r
## S3 method for class 'metadiag'
plot(x, parametric.smooth = TRUE, level = c(0.5, 0.75, 0.95), limits.x = c(0, 1), limits.y = c(0, 1), kde2d.n = 25,
     color.line = "red",
     title = paste("Posterior Predictive Contours (50%, 75% and 95%)"),
     marginals = TRUE, bin.hist = 30, color.hist = "lightblue",
     S = 500, color.pred.points = "lightblue",
     color.data.points = "blue", ...)
```

Arguments

- `x`: The object generated by the metadiag function.
- `parametric.smooth`: Indicates if the predictive curve is a parametric or non-parametric.
- `level`: Credibility levels of the predictive curve. If parametric.smooth = FALSE, then the probability levels are estimated from the nonparametric surface.
- `limits.x`: Numeric vector of length 2 specifying the x-axis limits. The default value is c(0, 1).
- `limits.y`: Numeric vector of length 2 specifying the x-axis limits. The default value is c(0, 1).
- `kde2d.n`: The number of grid points in each direction for the non-parametric density estimation. Can be scalar or a length-2 inter vector.
- `color.line`: Color of the predictive contour line.
- `title`: Optional parameter for setting a title in the plot.
- `marginals`: Plot the posterior marginal predictive histograms.
- `bin.hist`: Number of bins of the marginal histograms.
- `color.hist`: Color of the histograms.
- `S`: Number of predictive rates to be plotted.
- `color.pred.points`: Color of the posterior predictive rates.
color.data.points

Color of the data points.

See Also

metadiag.

Examples

```r
## Not run:
library(bandit)
data("glas")
glas.t <- glas[glas$marker == "Telomerase", 1:4]
glas.m1 <- metadiag(glas.t,               # Data frame
    re = "normal",                          # Random effects distribution
    re.model = "DS",                        # Random effects on D and S
    link = "logit",                         # Link function
    sd.Fisher.rho = 1.7,                    # Prior standard deviation of correlation
    nr.burnin = 1000,                       # Iterations for burnin
    nr.iterations = 10000,                  # Total iterations
    nr.chains = 2,                          # Number of chains
    r2jags = TRUE)                          # Use r2jags as interface to jags

plot(glas.m1,                               # Fitted model
    level = c(0.5, 0.75, 0.95),            # Credibility levels
    parametric.smooth = TRUE)             # Parametric curve

# Plot results: based on a non-parametric smoother of the posterior predictive rates .......

plot(glas.m1,                               # Fitted model
    level = c(0.5, 0.75, 0.95),            # Credibility levels
    parametric.smooth = FALSE)            # Non-parametric curve

# Using the pipe command in the package dplyr and changing some colors .......

library(dplyr)

glas.t %>%
    metadiag(re = "normal", re.model = "SeSp") %>%
    plot(parametric.smooth = FALSE,
          S = 100,
          color.data.points = "green",
          color.pred.points = "blue",
          color.line = "black")

## End(Not run)
```
Description

This function compares the predictive posterior surfaces of two fitted models.

Usage

plotcompare(m1, m2, level = 0.95, 
  title = paste("Comparative Predictive Posterior Contours"), 
  m1.name = "Model.1", m2.name = "Model.2", group = NULL, 
  limits.x = c(0, 1), limits.y = c(0, 1), group.colors = c("blue", "red"))

Arguments

  m1    A model fitted to the data. This is an object generated by the metadiag function.
  m2    A second model fitted to the data. This is an object generated by the metadiag function.
  level Credibility level of the predictive curves.
  title The title of the plot.
  m1.name Label of the model 1.
  m2.name Label of the model 2.
  group A factor variable to display data of different groups. The length of group must be the same as the total number of studies used to fit model 1 and model 2. For example, if 10 studies are used to fit model m1 and 5 studies are used to fit model m2, then the length(group)=15.
  limits.x A vector with the limits of the horizontal axis.
  limits.y A vector with the limits of the vertical axis.
  group.colors A character vector with two color names.

See Also

metadiag.

Examples

  ## execute analysis
  ## Not run:

  # Comparing results from two models same data

data(glas)
plotdata

Basic function to plot results of meta-analysis of diagnostic test data

**Description**

This function plots the true positive rates vs the false positive rates of each study included in the meta-analysis. Study results are displayed by circles, the diameter of each circle is proportional to the sample size of the study (or table). If subgroups are displayed each group is represented by different colours. This function use the package *ggplot2*.

**Usage**

```r
plotdata(data, two.by.two = FALSE, group = 1, x.lo = 0, x.up = 1, y.lo = 0, y.up = 1, alpha.p = 0.7, max.size = 15)
```

**Arguments**

- `data` Either a data frame with at least 4 columns containing the true positives (tp), number of patients with disease (n1), false positives (fp), number of patients without disease (n2), or for two.by.two = TRUE a data frame where each line contains the diagnostic results as a two by two table, where the column names are: TP, FP, TN, FN.
- `two.by.two` If TRUE indicates that the diagnostic results are given as: TP, FP, TN, FN.
**plotdata**

- `group`: a variable indicating a group factor
- `x.lo`: lower limit of the x-axis
- `x.up`: upper limit of the x-axis
- `y.lo`: lower limit of the y-axis
- `y.up`: upper limit of the y-axis
- `alpha.p`: transparency of the points
- `max.size`: scale parameter of the maximum size

**Examples**

```r
## execute analysis
## Not run:

data(ct)
gr <- with(ct, factor(d1,
labels = c("Retrospective study", "Prospective study")))

plotdata(ct,            # Data frame
  group = gr,          # Grouping variable
  y.lo = 0.75,         # Lower limit of y-axis
  x.up = 0.75,         # Upper limit of x-axis
  alpha.p = 0.5,       # Transparency of the balls
  max.size = 5)        # Scale the circles

plotdata(glas,         # Data frame
  group = glas$marker, # Grouping variable
  max.size = 5)        # Scale of circles

plotdata(scheidler,    # Data frame
  group = scheidler$test)

plotdata(safdar05)    # Data frame
plotdata(safdar05, group = safdar05$technique)

library(dplyr)
safdar05 %>% plotdata(group = safdar05$duration)

data(ep)
ep.gr <- with(ep, factor(d1,
ep.gr <- with(ep, factor(d1, labels = c("Prospective study", "Retrospective study")))

ep %>% plotdata(group = ep.gr)
```
ep %>% plotdata(group = factor(ep$nthres))

## End(Not run)

---

**plotesp**

*plotesp() plot the posterior densities for Se and Sp*

---

**Description**

plotesp() plot the posterior densities for Se and Sp

**Usage**

`plotesp(m, binwidth.p = 0.03, CI.level = 0.95)`

**Arguments**

- `m` The object generated by the metadiag function.
- `binwidth.p` Histograms binwidth, default is 0.03.
- `CI.level` Level of the posterior interval default is 0.95.

**See Also**

metadiag.

**Examples**

```r
## execute analysis
## Not run:
data(ep)
m1.ep <- metadiag(ep[,1:4])

plotesp(m = m1.ep)

## End(Not run)
```
Description

Conflict of evidence plot: this plot displays the posterior distribution of the study’s weights \( w_1 \) and \( w_1 \). These weights indicate potential conflict of evidence of the studies. The weight \( w_1 \) indicates deviations with respect to the specificity and \( w_2 \) to the sensitivity.

Usage

```r
plotw(m, group = NULL, group.colors = c("blue", "red"))
```

Arguments

- `m` the object generated by `metadiag`. The model object must be fitted with the options: `re = "sm"` and `split.w = TRUE`.
- `group` an optional argument which has to be a factor of the same length as the number of studies in the data. If set, then the plot is colored by groups.
- `group.colors` a character vector with two color names.

See Also

`metadiag`.

Examples

```r
## execute analysis
## Not run:
data(ep)
m.ep <- metadiag(ep[,1:4],
  re = "sm",
  re.model = "SeSp",
  split.w = TRUE,
  df.estimate = TRUE)

plotw(m.ep)

# Relationship between conflict and study design
plotw(m.ep, group = ep.gr)

## End(Not run)
```
print.metadiag

Generic print function for metadiag object in bamdit

Description

Generic print function for metadiag object in bamdit

Usage

```r
## S3 method for class 'metadiag'
print(x, digits = 3, ...)
```

Arguments

- `x`: The object generated by the function `metadiag`.
- `digits`: The number of significant digits printed. The default value is 3.
- `...`: Additional arguments.

safdar05

Diagnosis of Intravascular Device-Related Bloodstream Infection

Description

Outcome of individual studies evaluating intravascular device-related bloodstream infection

Format

A matrix with 78 rows and 8 columns. Each row represents study results, the columns are:

- `tp`: number of true positives.
- `n1`: number of patients with disease.
- `fp`: number of false positives.
- `n2`: number of patients without disease.
- `author`: first author of the study.
- `year`: publication date.
- `technique`: diagnostic technique used in the study.
- `duration`: duration of catheterization: short term or long term or both.

Source

The data were obtained from


Description

This data frame summarizes the tables 1-3 of Scheidler et al. 1997.

Format

A matrix with 46 rows and 7 columns. Each row represents study results, the columns are:

- tp  true positives.
- n1  number of patients with disease.
- fp  false positives.
- n2  number of patients without disease.
- author  first author of the study.
- year  publication date.
- test  test method used in the study.

Source

The data were obtained from


References


summary.metadiag

Generic summary function for metadiag object in bamdit

Description

Generic summary function for metadiag object in bamdit

Usage

```r
## S3 method for class 'metadiag'
summary(object, digits = 3, intervals = c(0.025, 0.5, 0.975), ...)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>The object generated by the metadiag function.</td>
</tr>
<tr>
<td>digits</td>
<td>The number of significant digits printed. The default value is 3.</td>
</tr>
<tr>
<td>intervals</td>
<td>A numeric vector of probabilities with values in [0,1]. The default value is intervals = c(0.025, 0.5, 0.975).</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Index

*Topic **Bayesian**
- bandit-package, 2

*Topic **Conflict**
- bandit-package, 2

*Topic **Detection,**
- bandit-package, 2

*Topic **Evidence,**
- bandit-package, 2

*Topic **Hierarchical**
- bandit-package, 2

*Topic **Meta-Analysis,**
- bandit-package, 2

*Topic **Modeling,**
- bandit-package, 2

*Topic **Multiparameters**
- bandit-package, 2

*Topic **Outliers**
- bandit-package, 2

*Topic **datasets**
  - ct, 4
  - ep, 5
  - glas, 6
  - Gould, 7
  - MRI, 14
  - safdar05, 22
  - scheidler, 23

*Topic **file**
  - bsroc, 3
  - plotcompare, 17
  - plotsesp, 20
  - plottw, 21

*Topic **of**
- bandit-package, 2

bamdit (bandit-package), 2
bamdit-package, 2
bsroc, 3

cp, 4

glas, 6

gould, 7

metadiag, 4, 7, 16, 17, 20, 21

mri, 14

plot.metadiag, 15
plotcompare, 17
plotdata, 18
plotdata2, 20
plotw, 21
print.metadiag, 22

safdar05, 22
scheidler, 23
summary.metadiag, 23

ct, 4