Package ‘lmviz’

May 15, 2019

Type Package

Title A Package to Visualize Linear Models Features and Play with Them

Version 0.1.2

Author Francesco Pauli (see file LICENSEMEDIA for credits on sounds and images)

Maintainer Francesco Pauli <`francesco.pauli@deams.units.it`>

Description Contains three shiny applications.
Two are meant to explore linear model inference feature through simulation.
The third is a game to learn interpreting diagnostic plots.

License GPL-3

Encoding UTF-8

LazyData true

Imports shiny, shinyjs, lmtest, mgcv, methods

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-15 14:30:03 UTC

R topics documented:

`BadLM`	*BadLM shiny app*

Description

Launches the BadLM shiny app, a tool to explore the consequences of the violation of homoscedasticity and/or normality assumptions in a linear model

Usage

```
BadLM(dist.custom = NULL, dist.custom.veravar = NULL,
  dist.custom.param = NULL)
```

Arguments

`dist.custom`	custom generator for Y, see examples below
`dist.custom.veravar`	variance function for dist.custom, see examples below
`dist.custom.param`	parameters for dist.custom, see examples below

Details

Allows to set a data generating mechanism for a response variable Y and an explanatory variable x such that $E(Y|X = x) = \beta_1 + \beta_2 x$, various possible distributions for Y are available, depending on the selected distributional assumptions the variance may also be set as a function of x. The program performs a number of simulations from the fit and visualizes the simulated sampling distributions of the estimators.

Full help is available from within the shiny app.

Value

None

Author(s)

Francesco Pauli, `<francesco.pauli@deams.units.it>`

Examples

```
## Not run:
if (interactive()){
BadLM()

# function to generate Y
dist=function(n,my,parvet,par,x) {
  my+parvet*rt(n,df=par[1])
}
# function to give the true value of the variance
varfun=function(my,parvet,par,x){
  if (par[1]>2) {
    veravar=parvet^2*par[1]/(par[1]-2)
  } else {
    veravar=-1
  }
  return(veravar)
}
# dist and varfun must have those argument where
#  my is the vector mean of Y
#  parvet is g() computed at x values
#  par is a vector of two parameters
param=list(nome="Student-t (bis)", #name of dist for drop down menu (optional)
           nomepar1="Gradi di libertà", #name of parameter 1 (optional)
           minpar1=1,maxpar1=30, #min/max of param 1 (needed)
           valuepar1=10, #initial value of param1 (optional)
           steppar1=0.1, #increment of param1 (optional)
           enableVarFunPanel=TRUE #whether the panel to input g should appear
)

BadLM(dist.custom=dist,dist.custom.veravar = varfun,dist.custom.param=param)


dist=function(n,my,parvet,par,x) {
 my+rnorm(n,0,sqrt(par[1]+par[2]*x^2))
}
# function to give the true value of the variance
varfun=function(my,parvet,par,x){
 return(par[1]+par[2]*x^2)
}
# dist and varfun must have those argument where
#  my is the vector mean of Y
#  parvet is g() computed at x values
#  par is a vector of two parameters
param=list(nome="N(.,b1+b2*x^2)", #name of dist for drop down menu (optional)
           nomepar1="b1", #name of parameter 1 (optional)
           minpar1=1,maxpar1=3, #min/max of param 1 (needed)
           valuepar1=1, #initial value of param1 (optional)
           steppar1=0.1, #increment of param1 (optional)
           nomepar2="b2", #name of parameter 1 (optional)
           minpar2=0,maxpar2=3, #min/max of param 1 (needed)
           valuepar2=1, #initial value of param1 (optional)
           steppar2=0.1, #increment of param1 (optional)
           enableVarFunPanel=FALSE, #whether the panel to input g should appear
           showVarFun=TRUE
)

  BadLM(dist.custom=dist,dist.custom.veravar = varfun,dist.custom.param=param)
}
## End(Not run)
```

checksim *Test the computer player performance*

Description

Assesses (by simulation) the performance of an algorithm for detecting non linearity/heteroscedasticity/non normality (`ComputerDecision.default`) on data generated by the function `Simulation` (`Simulation.default`).

Usage

```
checksim(m, ComputerDecision = ComputerDecision.default,
  Simulation = Simulation.default)
```

Arguments

`m`	number of simulations
`ComputerDecision`	function which returns a computer guess on the violation of assumption (see `ComputerDecision.default`)
`Simulation`	function which return a sample generated according to different assumptions (see `Simulation.default`)

Value

`table`	a 4x4 matrix, this is the frequency of the true data generating mechanism (rows) and the computer guess (column)

Author(s)

Francesco Pauli, `<francesco.pauli@deams.units.it>`

See Also

`ComputerDecision.default`,`Simulation.default`

Examples

```
checksim(10)
```

`ComputerDecision.default` *Computer player decision*

Description

Decides whether a fitted lm objects residuals are such that a violation of the assumptions of non linearity, heteroscedasticity, non normality occurs

Usage

```
ComputerDecision.default(fit)
```

Arguments

`fit`	an object returned by `lm`

Value

An integer between 1 and 4 where 1=non linearity; 2=heteroscedasticity; 3=non normality; 4=no violation

Author(s)

Francesco Pauli, `<francesco.pauli@deams.units.it>`

See Also

`Simulation.default`, `checksim`

Examples

```
x=rnorm(10)
y=x+rnorm(10,0,0.4)
fit=lm(y~x)
ComputerDecision.default(fit)

x=rnorm(30)
y=x+rt(30,2)
fit=lm(y~x)
ComputerDecision.default(fit)
```

`lmviz` *lmviz: A package to visualize linear models features and play with them.*

Description

The lmviz package contains three shyny apps

`SimpleLM` allows to input the parameters of a linear model and simulate from it, exploring the various diagnostics and sampling distributions

`BadLM` allows to set a data generating mechanism violating the homoscedasticity and/or normality assumptions of the linear model and explore by simulation its consequences on inference

`QuizResidual` a game in which the user is prompted to guess whether the diagnostic plots of a linear model suggest some hypotheses is violated

---

`QuizResidual` *QuizResidual shiny app*

Description

Launches the QuizResidual shiny app

Usage

```
QuizResidual(ComputerDecision = ComputerDecision.default,
  Simulation = Simulation.default, dir.images = NULL,
  dir.sounds = NULL)
```

Arguments

`ComputerDecision`	the function to be used to state the answer of the computer player (see `ComputerDecision.default`)
`Simulation`	the function to be used to simulate data (see `Simulation.default`)
`dir.images`	the directory where images to be used by the shiny app are to be found, default to NULL, images from the package will be used
`dir.sounds`	the directory where sounds to be used by the shiny app are to be found, default to NULL, sounds from the package will be used

Details

QuizResidual shiny app is a game in which the player is asked to guess, based on four standard diagnostic plots of a linear model, whether there is a violation of one of the basic assumptions: linearity, homoscedasticity, normality of errors.

The program will simulate a sample (x,Y) from a randomly chosen data generating mechanism possibly violating one of the assumptions (function `Simulation.default`), fit a linear model and plot the diagnostic.

The computer player makes a guess on whether there is a violation of assumptions (function `ComputerDecision.default`).

After the answer is given, the true data generating mechanism will be shown in the plot, in particular, the true regression function, the true standard deviation of errors and the true density of errors.

The game can be customized by coding your own Simulation and ComputerDecision functions passing them as arguments.

Sounds will be played depending on whether the correct or wrong answer is given and a final sound is played depending on the outcome, also an appropriate image is shown. (Sounds are taken from the site https://freesound.org/, images from https://wpclipart.com and are public domain, other sounds and images can be used by calling the app with the directories where the images are stored as argument, sounds must be named as follows: suonorr: sound to be played when both the player and the computer give the correct answer; suonowr: sound to be played when the player is correct and the computer is wrong; suonorw and suonoww are analogous; suonofinaleP/V/S: final sound to play in case of tie/win/loss; immagineP/V/S: image to be shown in case of tie/win/loss.)

Value

None

Author(s)

Francesco Pauli, <`francesco.pauli@deams.units.it`>

See Also

`ComputerDecision.default`, `Simulation.default`

Examples

```
## Not run:
if (interactive()){
  QuizResidual()
  # if custom sounds and images are in the directory www in the working directory
  QuizResidual(dir.images=paste0(getwd(),"/www"),dir.sounds=paste0(getwd(),"/www"))
}
## End(Not run)
```

SimpleLM *SimpleLM shiny app*

Description

Launches the SimpleLM shiny app, which allows to input the parameters of a linear model and simulate from it, exploring the various diagnostics and sampling distributions

Usage

```
SimpleLM()
```

Author(s)

Francesco Pauli, <francesco.pauli@deams.units.it>

Examples

```
## Not run:
if (interactive()){
 SimpleLM()
 }
## End(Not run)
```

Simulation.default *Simulates a sample*

Description

Simulates an (x,y) sample (suitable for estimating a lm) which can be either non linear/heteroscedastic/non normal or in line with standard lm assumptions

Usage

```
Simulation.default(model.to.sim)
```

Arguments

model.to.sim	an integer between 1 and 4 where 1=non linearity; 2=heteroscedasticity; 3=non normality; 4=no violation

Value

A list of objects (of which the first two are essential, the following are needed to display the correct solution in the shiny app)

`x,y`	the sample
`my`	the true mean of Y
`sderr`	the true standard deviation of errors
`errore`	the errors
`xperdens,ferrore`	coordinates of points of the true density of errors

Author(s)

Francesco Pauli, `<francesco.pauli@deams.units.it>`

See Also

`ComputerDecision.default`, `checksim`

Examples

```
Simulation.default(1)

Simulation.default(sample(1:4,1))
```

Index