Package 'PAmeasures'

January 20, 2025

Type Package

Title Prediction and Accuracy Measures for Nonlinear Models and for
Right-Censored Time-to-Event Data

Version 0.1.0

Date 2018-01-18

Author Xiaoyan Wang, Gang Li

Maintainer Xiaoyan Wang<xywang@ucla.edu>

Description We propose a pair of summary measures for the predictive power of a prediction
function based on a regression model. The regression model can be linear
or nonlinear, parametric, semi-parametric, or nonparametric, and correctly
specified or mis-specified. The first measure, R-squared, is an extension of
the classical R-squared statistic for a linear model, quantifying the prediction
function's ability to capture the variability of the response. The second
measure, L-squared, quantifies the prediction function's bias for predicting the
mean regression function. When used together, they give a complete summary of
the predictive power of a prediction function. Please refer to Gang Li and Xi-
aoyan Wang (2016) <arXiv:1611.03063> for more details.

Depends R (>= 3.1)

Imports survival, stats

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Suggests testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2018-01-22 10:06:41 UTC

Contents

---

moore   *Moore's Law data*

Description

A dataset containing the number of transistors and the corresponding #' years. The Moore's Law #'states that the number of transistors in a dense integrated circuit doubles approximately every two #'years. moore.

Usage

```
moore
```

Format

A data frame with 48 rows and 3 variables:

year year, from 1973 to 2011

time time starting from 1973

count number of transistors

---

pam.censor   *Prediction Accuracy Measures for Regression Models of Right-Censored Data*

Description

This function calculates a pair of measures, R-Squared and L-Squared, for any regression models of right-censored data. R-squared is an extension of the classical R^2 statistic for a linear model, quantifying the amount of variability in the response that is explained by a corrected prediction based on the original prediction function. L-squared is the proportion of the prediction error of the original prediction function that is explained by the corrected prediction function, quantifying the distance between the corrected and uncorrected predictions. When used together, they give a complete summary of the predictive power of a prediction function.

Usage

```
pam.censor(y, y.predict, delta)
```

Arguments

y	A numeric vector containing the response values.
y.predict	A numeric vector containing the predicted response values from a fitted model.
delta	A numeric vector indicating the status of the event, normally 0=alive, 1=dead.

Value

A list containing two components: R-squared and L-squared

Examples

```
library(survival)
library(PAmeasures)

# Use Mayo Clinic Primary Biliary Cirrhosis Data
data(pbc)

# Fit an exponential model with bilirubin
fit1 <- survreg(Surv(time, status==2) ~ bili, data = pbc,dist="exponential" )

# Obtain predicted response from the fitted exponential model
predict.time<-predict(fit1,type="response")

# Recode status at endpoint, 0 for censored, 1 for dead
delta.pbc<- as.numeric(pbc$status == 2)

# R.squared and L.squared of log-linear model
pam.censor(pbc$time, predict.time, delta.pbc)
```

---

pam.coxph	*Prediction Accuracy Measures for Cox proportional hazards model*

Description

This function calculates a pair of measures, R-Squared and L-Squared, for Cox proportional hazards model. R-squared is an extension of the classical R2 statistic for a linear model, quantifying the amount of variability in the response that is explained by a corrected prediction based on the original prediction function. L-squared is the proportion of the prediction error of the original prediction function that is explained by the corrected prediction function, quantifying the distance between the corrected and uncorrected predictions. When used together, they give a complete summary of the predictive power of a prediction function.

Usage

```
pam.coxph(fit.cox)
```

Arguments

fit.cox   object inheriting from class coxph representing a fitted Cox proportional hazards
                       regression model. Specifying x = TRUE and y=TRUE are required in the call to
                       coxph( ) to include the design matrix and the response vector in the object fit.

Value

A list containing two components: R-squared and L-squared

Examples

```
library(survival)
library(PAmeasures)

# Use Mayo Clinic Primary Biliary Cirrhosis Data
data(pbc)

head(pbc)


# Fit a univariate Cox PH model with standardised blood clotting time
fit1 <- coxph(Surv(time, status==2) ~ protime, data = pbc,x=TRUE,y=TRUE)

# R.squared and L.squared of Cox PH model
pam.coxph(fit1)

# Fit a multiple Cox PH model with bilirubin and standardised blood clotting time
fit2 <- coxph(Surv(time, status==2) ~ bili + protime, data = pbc,x=TRUE,y=TRUE)

# R.squared and L.squared of Cox PH model
pam.coxph(fit2)
```

---

pam.nlm   *Prediction Accuracy Measures for Nonlinear Regression Models.*

Description

This function calculates a pair of measures, R-Squared and L-Squared, for any nonlinear regression
model. R-squared is an extension of the classical R^2 statistic for a linear model, quantifying the
amount of variability in the response that is explained by a corrected prediction based on the original
prediction function. L-squared is the proportion of the prediction error of the original prediction
function that is explained by the corrected prediction function, quantifying the distance between the
corrected and uncorrected predictions. When used together, they give a complete summary of the
predictive power of a prediction function.

Usage

```
pam.nlm(y, y.predict)
```

Arguments

y   A numeric vector containing the response values.

y.predict   A numeric vector containing the predicted response values from a fitted model.

Value

A list containing two components: R-squared and L-squared

Examples

```
library(PAmeasures)

data(moore)

head(moore)

# Transistor count
count <- moore$count

time<-moore$time

# Fit a log-linear model
moore.glm= glm(log2(count) ~ time, family=gaussian(link = "identity") )

# Obtain predicted transistor count
count.predict<-2^(predict(moore.glm,newdata = data.frame(X = time),type = "response" ))

# R.squared and L.squared of log-linear model
pam.nlm(count, count.predict)
```

---

pam.survreg   *Prediction Accuracy Measures for Parametric Survival Regression Models*

Description

This function calculates a pair of measures, R-Squared and L-Squared, for parametric survival regression models. R-squared is an extension of the classical R2 statistic for a linear model, quantifying the amount of variability in the response that is explained by a corrected prediction based on the original prediction function. L-squared is the proportion of the prediction error of the original prediction function that is explained by the corrected prediction function, quantifying the distance between the corrected and uncorrected predictions. When used together, they give a complete summary of the predictive power of a prediction function.

Usage

```
pam.survreg(fit.survreg)
```

Arguments

fit.survreg   object inheriting from class survreg representing a fitted parametric survival regression model. Specifying x = TRUE and y=TRUE are required in the call to survreg( ) to include the design matrix and the response vector in the object fit.

Value

A list containing two components: R-squared and L-squared

Examples

```
library(survival)
library(PAmeasures)

# Use Mayo Clinic Primary Biliary Cirrhosis Data
data(pbc)

head(pbc)

# Fit an exponential model with bilirubin
fit1 <- survreg(Surv(time, status==2) ~ bili, data = pbc,dist="exponential",x=TRUE,y=TRUE)

# R.squared and L.squared of exponential model
pam.survreg(fit1)

# Fit a lognormal model with standardised blood clotting time
fit2 <- survreg(Surv(time, status==2) ~ protime, data = pbc,dist="lognormal",x=TRUE,y=TRUE)

# R.squared and L.squared of lognormal model
pam.survreg(fit2)

# Fit a weibull model with bilirubin and standardised blood clotting time
fit3 <- survreg(Surv(time, status==2) ~ bili + protime, data = pbc,dist="weibull",x=TRUE,y=TRUE)

# R.squared and L.squared of weibull model
pam.survreg(fit3)
```

Index