Package: predieval (via r-universe)

October 13, 2024

Type Package

Title Assessing Performance of Prediction Models for Predicting Patient-Level Treatment Benefit

Version 0.1.2

Date 2022-11-01

Author Orestis Efthimiou

Maintainer Orestis Efthimiou <oremiou@gmail.com>

Description Methods for assessing the performance of a prediction model with respect to identifying patient-level treatment benefit. All methods are applicable for continuous and binary outcomes, and for any type of statistical or machine-learning prediction model as long as it uses baseline covariates to predict outcomes under treatment and control.

License GPL (>= 2)

Depends R (>= 4.1)

Imports stats, Hmisc (>= 4.6-0), ggplot2 (>= 3.3.5), MASS (>= 7.3), Matching (>= 4.10-2)

Encoding UTF-8

URL https://github.com/esm-ispm-unibe-ch/predieval

LazyData true

RoxygenNote 7.1.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Repository https://esm-ispm-unibe-ch.r-universe.dev

RemoteUrl https://github.com/esm-ispm-unibe-ch/predieval

RemoteRef HEAD

RemoteSha 225b2573be9bdfba00fbb8345ff455043f75a25d

Contents

`bencalibr` *Plotting calibration for benefit of a prediction model*

Description

This function produces a plot to illustrate the calibration for benefit for a prediction model. The samples are split into a number of groups according to their predicted benefit, and within each group the function estimates the observed treatment benefit and compares it with the predicted one

Usage

```
bencalibr(
  data = NULL,
  Ngroups = 5,
  y.observed,
  treat,
  predicted.treat.0,
  predicted.treat.1,
  type = "continuous",
  smoothing.function = "lm",
  axis.limits = NULL
)
```

Arguments

`data`	An optional data frame containing the required information.
`Ngroups`	The number of groups to split the data.
`y.observed`	The observed outcome.
`treat`	A vector with the treatment assignment. This must be 0 (for control treatment) or 1 (for active treatment).
`predicted.treat.0`	A vector with the model predictions for each patient, under the control treatment. For the case of a binary outcome this should be probabilities of an event.
`predicted.treat.1`	A vector with the model predictions for each patient, under the active treatment. For the case of a binary outcome this should be probabilities of an event.

`type`	The type of the outcome, "binary" or "continuous".
`smoothing.function`	The method used to smooth the calibration line. Can be "lm", "glm", "gam", "loess", "rlm". More details can be found in https://ggplot2.tidyverse.org/reference/geom_smooth.html.
`axis.limits`	Sets the limits of the graph. It can be a vector of two values, i.e. the lower and upper limits for x and y axis. It can be omitted.

Value

The calibration plot

Examples

```
# continuous outcome
dat1=simcont(200)$dat
head(dat1)
lm1=lm(y.observed~(x1+x2+x3)*t, data=dat1)
dat.t0=dat1; dat.t0$t=0
dat.t1=dat1; dat.t1$t=1
dat1$predict.treat.1=predict(lm1, newdata = dat.t1) # predictions in treatment
dat1$predict.treat.0=predict(lm1, newdata = dat.t0) # predicions in control
bencalibr(data=dat1, Ngroups=10, y.observed, predicted.treat.1=predict.treat.1,
          predicted.treat.0=predict.treat.0, type="continuous", treat=t,
          smoothing.function = "lm", axis.limits = c(-2, 2))
# binary outcome
dat2=simbinary(500)$dat
head(dat2)
glm1=glm(y.observed~(x1+x2+x3)*t, data=dat2, family = binomial(link = "logit"))
dat2.t0=dat2; dat2.t0$t=0
dat2.t1=dat2; dat2.t1$t=1
dat2$predict.treat.1=predict(glm1, newdata = dat2.t1) # predictions in treatment
dat2$predict.treat.0=predict(glm1, newdata = dat2.t0) # predicions in control
bencalibr(data=dat2, Ngroups=6, y.observed, predicted.treat.1=expit(predict.treat.1),
          predicted.treat.0=expit(predict.treat.0), type="binary", treat=t,
          smoothing.function = "lm")
```

`expit` *Expit*

Description

Calculates the expit of a real number

Usage

```
expit(x)
```

Arguments

`x`	A real number

Value

exp(x)/(1+exp(x))

Examples

```
expit(2.3)
```

`logit` *Logit*

Description

Calculates the logit of a real number between 0 and 1

Usage

```
logit(x)
```

Arguments

`x`	A real number between 0 and 1

Value

log(x/(1-x))

Examples

```
logit(0.2)
```

`predieval` *Calculating measures for calibration for benefit for a prediction model*

Description

This function calculates a series of measures to assess decision accuracy, discrimination for benefit, and calibration for benefit of a prediction model.

Usage

```
predieval(
  repeats = 50,
  Ngroups = 10,
  X,
  treat,
  Y,
  predicted.treat.1,
  predicted.treat.0,
  type = "continuous",
  bootstraps = 500,
  Threshold = 0
)
```

Arguments

`repeats`	The number of repetitions for the algorithm.
`Ngroups`	The number of groups to split the data.
`X`	A dataframe with patient covariates.
`treat`	A vector with the treatment assignment. This must be 0 (for control treatment) or 1 (for active treatment).
`Y`	The observed outcome. For binary outcomes this should be 0 or 1
`predicted.treat.1`	A vector with the model predictions for each patient, under the active treatment. For the case of a binary outcome this should be probabilities of an event.
`predicted.treat.0`	A vector with the model predictions for each patient, under the control treatment. For the case of a binary outcome this should be probabilities of an event.
`type`	The type of the outcome, "binary" or "continuous".
`bootstraps`	The number of bootstrap samples to be used for calculating confidence intervals.
`Threshold`	Threshold for treatment benefit

Value

A table with all estimated measures of performance.

Examples

```
# continuous outcome
dat0=simcont(500)$dat
head(dat0)
# Randomly shuffle the data
dat<-dat0[sample(nrow(dat0)),]
# Create random folds
dat$folds <- cut(seq(1,nrow(dat)),breaks=10,labels=FALSE)

# Obtain out-of-sample predictions
```

```
dat.out.CV<-list()
for (i in 1:10){
  dat.in.CV=dat[dat$folds!=i,]
  dat.out.CV[[i]]=dat[dat$folds==i,]
  dat1<-dat.out.CV[[i]]; dat1$t=1
  dat0<-dat.out.CV[[i]]; dat0$t=0
  m1=lm(data=dat.in.CV, y.observed~x1*t+x2*t)
  dat.out.CV[[i]]$predict.treat.1=predict(newdata=dat1, m1)# predictions in treatment
  dat.out.CV[[i]]$predict.treat.0=predict(newdata=dat0, m1)# predicions in control
}

dat.CV=dat.out.CV[[1]]
for (i in 2:10){  dat.CV=rbind(dat.CV,dat.out.CV[[i]])}

# assess model performance
predieval(repeats=20, Ngroups=c(5:10),
            X=dat.CV[,c("x1", "x2","x3")],
            Y=dat.CV$y.observed,
            predicted.treat.1 = dat.CV$predict.treat.1,
            predicted.treat.0 = dat.CV$predict.treat.0,
            treat=dat.CV$t, type="continuous")


# binary outcome
dat0=simbinary(500)$dat
head(dat0)

# Randomly shuffle the data
dat<-dat0[sample(nrow(dat0)),]
# Create random folds
dat$folds <- cut(seq(1,nrow(dat)),breaks=10,labels=FALSE)

dat.out.CV<-list()
for (i in 1:10){
  dat.in.CV=dat[dat$folds!=i,]
  dat.out.CV[[i]]=dat[dat$folds==i,]
  dat1<-dat.out.CV[[i]]; dat1$t=1
  dat0<-dat.out.CV[[i]]; dat0$t=0
  glm1=glm(y.observed~(x1+x2+x3)*t, data=dat.in.CV, family = binomial(link = "logit"))
  dat.out.CV[[i]]$predict.treat.1=predict(newdata=dat1, glm1)# predictions in treatment
  dat.out.CV[[i]]$predict.treat.0=predict(newdata=dat0, glm1)# predicions in control
}

dat.CV=dat.out.CV[[1]]
for (i in 2:10){  dat.CV=rbind(dat.CV,dat.out.CV[[i]])}


predieval(repeats=20, Ngroups=c(5:10), X=dat.CV[,c("x1", "x2","x3")],
            Y=dat.CV$y.observed,
            predicted.treat.1 = expit(dat.CV$predict.treat.1),
            predicted.treat.0 = expit(dat.CV$predict.treat.0),
            treat=dat.CV$t, type="binary",bootstraps = 50)
```

`simbinary` *Simulate data for a binary outcome*

Description

This function generates a dataframe with 6 patient covariates and a binary outcome simulated from a model that uses the covariates.

Usage

```
simbinary(Npat = 100)
```

Arguments

`Npat`	Number of patients to simulate.

Value

The function returns a dataframe with:

x1, x2, x3, x4: patient covariates.

t= treatment assignment (0 for control, 1 for active).

logit.control= the logit of the probability of an outcome in the control treatment.

logit.active= the logit of the probability of an outcome in the active treatment.

benefit= treatment benefit in log odds ratio.

py=the probability of the outcome for each patient, under the treatment actually administered.

logit.py= the logit of py.

y.observed= the observed outcome

Examples

```
dat1=simbinary(100)$dat
head(dat1)
```

`simcont` *Simulate data for a prediction model of a continuous outcome*

Description

This function generates a dataframe with 6 patient covariates and a continuous outcome simulated from a model that uses the covariates.

Usage

```
simcont(Npat = 100)
```

Arguments

`Npat`	Number of patients to simulate.

Value

The function returns a dataframe with:

x1, x2, x3, x4: patient covariates.

t= treatment assignment (0 for control, 1 for active).

y.control= the outcome if the patient takes the control treatment.

y.active= the outcome if the patient takes the active treatment.

benefit= the treatment benefit, i.e. y.active-y.control.

y.observed= the observed outcome.

Examples

```
dat1=simcont(100)$dat
head(dat1)
```

Index