Package ‘sleev’

May 25, 2022

Type Package
Title Semiparametric Likelihood Estimation with Errors in Variables
Version 1.0.2
Description Efficient regression analysis under general two-phase sampling, where Phase I includes error-prone data and Phase II contains validated data on a subset.
License GPL (>= 2)
Encoding UTF-8
RoxygenNote 7.2.0
Depends Rcpp (>= 1.0.7), R (>= 3.5.0)
LinkingTo Rcpp, RcppArmadillo, RcppEigen
Suggests knitr, lme4, MASS, rmarkdown, splines, testthat, R.rsp
VignetteBuilder R.rsp
LazyData true
NeedsCompilation yes
Author Sarah Lotspeich [aut],
Ran Tao [aut, cre],
Joey Sherrill [prg]
Maintainer Ran Tao <r.tao@vanderbilt.edu>
Repository CRAN
Date/Publication 2022-05-24 23:40:06 UTC

R topics documented:
cv_linear2ph ................................................................. 2
linear2ph ................................................................. 4
logistic2ph ............................................................... 7
mock.vccc ............................................................... 9

Index

1
cv_linear2ph

Perform cross-validation to calculate the average predicted log likelihood for the linear2ph function. This function can be used to select the B-spline basis that yields the largest average predicted log likelihood.

Description

Performs cross-validation to calculate the average predicted log likelihood for the linear2ph function. This function can be used to select the B-spline basis that yields the largest average predicted log likelihood.

Usage

cv_linear2ph(
    Y_unval = NULL,
    Y = NULL,
    X_unval = NULL,
    X = NULL,
    Z = NULL,
    Bspline = NULL,
    data = NULL,
    nfolds = 5,
    MAX_ITER = 2000,
    TOL = 1e-04,
    verbose = FALSE
)

Arguments

Y_unval
Specifies the column of the error-prone outcome that is continuous. Subjects with missing values of Y_unval are omitted from the analysis. This argument is required.

Y
Specifies the column that stores the validated value of Y_unval in the second phase. Subjects with missing values of Y are considered as those not selected in the second phase. This argument is required.

X_unval
Specifies the columns of the error-prone covariates. Subjects with missing values of X_unval are omitted from the analysis. This argument is required.

X
Specifies the columns that store the validated values of X_unval in the second phase. Subjects with missing values of X are considered as those not selected in the second phase. This argument is required.

Z
Specifies the columns of the accurately measured covariates. Subjects with missing values of Z are omitted from the analysis. This argument is optional.

Bspline
Specifies the columns of the B-spline basis. Subjects with missing values of Bspline are omitted from the analysis. This argument is required.
data Specifies the name of the dataset. This argument is required.
nfolds Specifies the number of cross-validation folds. The default value is 5. Although nfolds can be as large as the sample size (leave-one-out cross-validation), it is not recommended for large datasets. The smallest value allowable is 3.
MAX_ITER Specifies the maximum number of iterations in the EM algorithm. The default number is 2000. This argument is optional.
TOL Specifies the convergence criterion in the EM algorithm. The default value is 1E-4. This argument is optional.
verbose If TRUE, then show details of the analysis. The default value is FALSE.

Value

avg_pred_loglike Stores the average predicted log likelihood.
pred_loglike Stores the predicted log likelihood in each fold.
converge Stores the convergence status of the EM algorithm in each run.

Examples

rho = 0.3
p = 0.3
n = 100
n2 = 40
alpha = 0.3
beta = 0.4

### generate data
simX = rnorm(n)
epsilon = rnorm(n)
simY = alpha+beta*simX+epsilon
error = MASS::mvrnorm(n, mu=c(0,0), Sigma=matrix(c(1, rho, rho, 1), nrow=2))

simS = rbinom(n, 1, p)
simu = simS*error[,2]
simW = simS*error[,1]
simY_tilde = simY+simW
simX_tilde = simX+simU

id_phase2 = sample(n, n2)

simY[-id_phase2] = NA
simX[-id_phase2] = NA

# cubic basis
nsieves = c(5, 10)
pred_loglike = rep(NA, length(nsieves))
for (i in 1:length(nsieves)) {
  nsieve = nsieves[i]
  Bspline = splines::bs(simX_tilde, df=nsieve, degree=3,
                        Boundary.knots=range(simX_tilde), intercept=TRUE)
colnames(Bspline) = paste("bs", 1:nsieve, sep="")
# cubic basis

data = data.frame(Y_tilde=simY_tilde, X_tilde=simX_tilde, Y=simY, X=simX, Bspline)
### generate data

res = cv_linear2ph(Y="Y", X="X", Y_unval="Y_tilde", X_unval="X_tilde",
                  Bspline=colnames(Bspline), data=data, nfolds = 5)
pred_loglike[i] = res$avg_pred_loglik
}
data.frame(nsieves, pred_loglike)

---

linear2ph

Sieve maximum likelihood estimator (SMLE) for two-phase linear regression problems

Description

Performs efficient semiparametric estimation for general two-phase measurement error models when there are errors in both the outcome and covariates.

Usage

linear2ph(
  Y_unval = NULL,
  Y = NULL,
  X_unval = NULL,
  X = NULL,
  Z = NULL,
  Bspline = NULL,
  data = NULL,
  hn_scale = 1,
  noSE = FALSE,
  TOL = 1e-04,
  MAX_ITER = 1000,
  verbose = FALSE
)

Arguments

Y_unval Column name of the error-prone or unvalidated continuous outcome. Subjects with missing values of Y_unval are omitted from the analysis. This argument is required.

Y Column name that stores the validated value of Y_unval in the second phase. Subjects with missing values of Y are considered as those not selected in the second phase. This argument is required.
linear2ph

X_unval Specifies the columns of the error-prone covariates. Subjects with missing values of X_unval are omitted from the analysis. This argument is required.

X Specifies the columns that store the validated values of X_unval in the second phase. Subjects with missing values of X are considered as those not selected in the second phase. This argument is required.

Z Specifies the columns of the accurately measured covariates. Subjects with missing values of Z are omitted from the analysis. This argument is optional.

Bspline Specifies the columns of the B-spline basis. Subjects with missing values of Bspline are omitted from the analysis. This argument is optional.

data Specifies the name of the dataset. This argument is required.

hn_scale Specifies the scale of the perturbation constant in the variance estimation. For example, if hn_scale = 0.5, then the perturbation constant is 0.5 \( n^{-1/2} \), where \( n \) is the first-phase sample size. The default value is 1. This argument is optional.

noSE If TRUE, then the variances of the parameter estimators will not be estimated. The default value is FALSE. This argument is optional.

TOL Specifies the convergence criterion in the EM algorithm. The default value is 1E-4. This argument is optional.

MAX_ITER Maximum number of iterations in the EM algorithm. The default number is 1000. This argument is optional.

verbose If TRUE, then show details of the analysis. The default value is FALSE.

Value

coefficients Stores the analysis results.

sigma Stores the residual standard error.

covariance Stores the covariance matrix of the regression coefficient estimates.

converge In parameter estimation, if the EM algorithm converges, then converge = TRUE. Otherwise, converge = FALSE.

converge_cov In variance estimation, if the EM algorithm converges, then converge_cov = TRUE. Otherwise, converge_cov = FALSE.

References


See Also

cv_linear2ph() to calculate the average predicted log likelihood of this function.
Examples

\[
\begin{align*}
\text{rho} &= -.3 \\
\text{p} &= 0.3 \\
\text{hn_scale} &= 1 \\
\text{nsieve} &= 20
\end{align*}
\]

\[
\begin{align*}
n &= 100 \\
n_2 &= 40 \\
\alpha &= 0.3 \\
\beta &= 0.4 \\
\text{set.seed}(12345)
\end{align*}
\]

### generate data

\[
\begin{align*}
simX &= \text{rnorm}(n) \\
\text{epsilon} &= \text{rnorm}(n) \\
simY &= \alpha + \beta \times \text{simX} + \text{epsilon} \\
\text{error} &= \text{MASS::mvrnorm}(n, \text{mu} = \text{c}(0,0), \text{Sigma} = \text{matrix}(\text{c}(1, \text{rho}, \text{rho}, 1), \text{nrow} = 2))
\end{align*}
\]

\[
\begin{align*}
simS &= \text{rbinom}(n, 1, p) \\
simU &= \text{simS} * \text{error[,2]} \\
simW &= \text{simS} * \text{error[,1]} \\
simY_tilde &= \text{simY} + \text{simW} \\
simX_tilde &= \text{simX} + \text{simU}
\end{align*}
\]

\[
\begin{align*}
id\_phase2 &= \text{sample}(n, n_2)
\end{align*}
\]

\[
\begin{align*}
simY[-\text{id\_phase2}] &= \text{NA} \\
simX[-\text{id\_phase2}] &= \text{NA}
\end{align*}
\]

### histogram basis

\[
\begin{align*}
\text{Bspline} &= \text{matrix}(\text{NA}, \text{nrow} = n, \text{ncol} = \text{nsieve}) \\
\text{cut\_x\_tilde} &= \text{cut}(\text{simX\_tilde}, \text{breaks} = \text{quantile}(\text{simX\_tilde}, \text{probs} = \text{seq}(0, 1, 1/\text{nsieve})), \text{include.lowest} = \text{TRUE}) \\
\text{for} (i \text{ in} 1:\text{nsieve}) \{ \\
\text{Bspline[,i]} &= \text{as.numeric}(\text{cut\_x\_tilde} == \text{names}(\text{table}(\text{cut\_x\_tilde}))[i]) \\
\} \\
\text{colnames}(\text{Bspline}) &= \text{paste}("bs", 1:\text{nsieve}, \text{sep}="")
\end{align*}
\]

### linear basis

\[
\begin{align*}
\text{Bspline} &= \text{splines::bs}(\text{simX\_tilde}, \text{df} = \text{nsieve}, \text{degree} = 1, \\
\text{Boundary.knots} = \text{range}(\text{simX\_tilde}), \text{intercept} = \text{TRUE}) \\
\text{colnames}(\text{Bspline}) &= \text{paste}("bs", 1:\text{nsieve}, \text{sep}="")
\end{align*}
\]

### quadratic basis

\[
\begin{align*}
\text{Bspline} &= \text{splines::bs}(\text{simX\_tilde}, \text{df} = \text{nsieve}, \text{degree} = 2, \\
\text{Boundary.knots} = \text{range}(\text{simX\_tilde}), \text{intercept} = \text{TRUE}) \\
\text{colnames}(\text{Bspline}) &= \text{paste}("bs", 1:\text{nsieve}, \text{sep}="")
\end{align*}
\]

### cubic basis
Bspline = splines::bs(simX_tilde, df=nsieve, degree=3, Boundary.knots=range(simX_tilde), intercept=TRUE)
colnames(Bspline) = paste("bs", 1:nsieve, sep="")  # cubic basis

data = data.frame(Y_tilde=simY_tilde, X_tilde=simX_tilde, Y=simY, X=simX, Bspline)
res = linear2ph(Y="Y", X="X", Y_unval="Y_tilde", X_unval="X_tilde",
                Bspline=colnames(Bspline), data=data, hn_scale=0.1)

logistic2ph

Sieve maximum likelihood estimator (SMLE) for two-phase logistic regression problems

Description

This function returns the sieve maximum likelihood estimators (SMLE) for the logistic regression model from Lotspeich et al. (2021).

Usage

logistic2ph(Y_unval = NULL, Y = NULL, X_unval = NULL, X = NULL, Z = NULL, Bspline = NULL, data = NULL, hn_scale = 1, noSE = FALSE, TOL = 1e-04, MAX_ITER = 1000, verbose = FALSE)

Arguments

Y_unval Column name of the error-prone or unvalidated binary outcome. This argument is required.
Y Column name that stores the validated value of Y_unval in the second phase. Subjects with missing values of Y are considered as those not selected in the second phase. This argument is required.
X_unval Specifies the columns of the error-prone covariates. This argument is required.
X Specifies the columns that store the validated values of X_unval in the second phase. Subjects with missing values of X are considered as those not selected in the second phase. This argument is required.
Z Specifies the columns of the accurately measured covariates. This argument is optional.

Bspline Specifies the columns of the B-spline basis. This argument is required.

data Specifies the name of the dataset. This argument is required.

hn_scale Specifies the scale of the perturbation constant in the variance estimation. For example, if \( hn\_scale = 0.5 \), then the perturbation constant is \( 0.5n^{-1/2} \), where \( n \) is the first-phase sample size. The default value is 1. This argument is optional.

noSE If TRUE, then the variances of the parameter estimators will not be estimated. The default value is FALSE. This argument is optional.

TOL Specifies the convergence criterion in the EM algorithm. The default value is \( 1E-4 \). This argument is optional.

MAX_ITER Maximum number of iterations in the EM algorithm. The default number is 1000. This argument is optional.

verbose If TRUE, then show details of the analysis. The default value is FALSE.

Value

coefficients Stores the analysis results.

outcome_err_coefficients Stores the outcome error model results.

Bspline_coefficients Stores the final B-spline coefficient estimates.

covariance Stores the covariance matrix of the regression coefficient estimates.

converge In parameter estimation, if the EM algorithm converges, then converge = TRUE. Otherwise, converge = FALSE.

converge_cov In variance estimation, if the EM algorithm converges, then converge_cov = TRUE. Otherwise, converge_cov = FALSE.

converge_msg In parameter estimation, if the EM algorithm does not converge, then converged_msg is a string description.

References


Examples

```r
set.seed(918)

# Set sample sizes ----------------------------------------
N <- 1000 # Phase-I = N
n <- 250 # Phase-II/audit size = n

# Generate true values Y, Xb, Xa -------------------------
Xa <- rbinom(n = N, size = 1, prob = 0.25)
```

Xb <- rbinom(n = N, size = 1, prob = 0.5)
Y <- rbinom(n = N, size = 1, prob = (1 + exp(-(-0.65 - 0.2 * Xb - 0.1 * Xa))) ^ (-1))

# Generate error-prone Xb* from error model P(Xb*|Xb,Xa) --
sensX <- specX <- 0.75
delta0 <- - log(specX / (1 - specX))
delta1 <- - delta0 - log((1 - sensX) / sensX)
Xbstar <- rbinom(n = N, size = 1,
    prob = (1 + exp(- (delta0 + delta1 * Xb + 0.5 * Xa))) ^ (-1))

# Generate error-prone Y* from error model P(Y*|Xb*,Y,Xb,Xa)
sensY <- 0.95
specY <- 0.90
theta0 <- - log(specY / (1 - specY))
theta1 <- - theta0 - log((1 - sensY) / sensY)
Ystar <- rbinom(n = N, size = 1,
    prob = (1 + exp(- (theta0 - 0.2 * Xbstar + theta1 * Y - 0.2 * Xb - 0.1 * Xa))) ^ (-1))

## V is a TRUE/FALSE vector where TRUE = validated -------
V <- seq(1, N) %in% sample(x = seq(1, N), size = n, replace = FALSE)

# Build dataset --------------------------------------------
sdat <- cbind(id = 1:N, Y, Xb, Ystar, Xbstar, Xa)
# Make Phase-II variables Y, Xb NA for unaudited subjects ---
sdat[!V, c("Y", "Xb")]<- NA

# Fit model -----------------------------------------------
### Construct B-spline basis -------------------------------
### Since Xb* and Xa are both binary, reduces to indicators --
nsieve <- 4
B <- matrix(0, nrow = N, ncol = nsieve)
B[which(Xa == 0 & Xbstar == 0), 1] <- 1
B[which(Xa == 0 & Xbstar == 1), 2] <- 1
B[which(Xa == 1 & Xbstar == 0), 3] <- 1
B[which(Xa == 1 & Xbstar == 1), 4] <- 1
colnames(B) <- paste0("bs", seq(1, nsieve))
sdat <- cbind(sdat, B)
smle <- logistic2ph(Y_unval = "Ystar",
    Y = "Y",
    X_unval = "Xbstar",
    X = "Xb",
    Z = "Xa",
    Bspline = colnames(B),
    data = sdat,
    noSE = FALSE,
    MAX_ITER = 1000,
    TOL = 1E-4)

---

Mock VCCC dataset.
Description

A simulated dataset constructed to imitate the Vanderbilt Comprehensive Care Clinic (VCCC) patient records, which have been fully validated and therefore contain validated and unvalidated versions of all variables. The VCCC cohort is a good candidate for the purpose of illustration. The data presented in this section are a mocked-up version of the actual data due to confidentiality, but the data structure and features, such as mean and variability, closely resemble the real dataset.

Usage

mock.vccc

Format

A data frame with 2087 rows and 8 variables:

- **VL_unval**: viral load at antiretroviral therapy (ART) initiation, error-prone outcome, continuous
- **VL_val**: viral load at antiretroviral therapy (ART) initiation, validated outcome, continuous
- **ADE_unval**: having an AIDS-defining event (ADE) within one year of ART initiation, error-prone outcome, binary
- **ADE_val**: having an AIDS-defining event (ADE) within one year of ART initiation, validated outcome, binary
- **CD4_unval**: CD4 count at ART initiation, error-prone covariate, continuous
- **CD4_val**: CD4 count at ART initiation, validated covariate, continuous
- **ART**: whether patient is ART naive at enrollment, error-free covariate, binary
- **Gender**: gender of patient, 1 indicates male and 0 indicates female & error-free covariate, binary
Index

* datasets
  mock.vccc, 9

  cv_linear2ph, 2
  cv_linear2ph(), 5

  linear2ph, 4
  logistic2ph, 7
  mock.vccc, 9