Package 'flexPM'

November 23, 2015

Type Package

Title Flexible Parametric Models for Censored and Truncated Data

Version 2.0

Date 2015-11-21

Author Paolo Frumento <paolo.frumento@ki.se>

Maintainer Paolo Frumento <paolo.frumento@ki.se>

Description Estimation of flexible parametric models for survival data.

Depends survival

Imports splines, stats

License GPL-2

RoxygenNote 5.0.0

NeedsCompilation no

Repository CRAN

Date/Publication 2015-11-23 18:30:43

R topics documented:

`flexPM` *Flexible Modeling of Survival Data*

Description

Fit a flexible parametric regression model to possibly right-censored, left-truncated data.

Usage

```
flexPM(formula, data, weights, df = 3, degree = 3, knots, maxit, tol = 1e-6)
```

Arguments

formula	an object of class "formula": a symbolic description of the model to be fitted. The response must be a survival object as returned by the Surv function. See 'Details'.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If missing, the variables are taken from environment(formula).
weights	optional weights to be used during the fitting process.
df	the degrees of freedom of the B-spline that describes $u(T)$ (see 'Details').
degree	the degree of the polynomial of the B-spline that describes $u(T)$.
knots	the *internal* knots of the B-spline that describes $u(T)$.
maxit	maximum number of iterations of the Newton-Raphson algorithm. If missing, a default value based on the number of free parameters is used.
tol	tolerance for the Newton-Raphson algorithm.

Details

This function fits a flexible parametric model to possibly right-censored, left-truncated outcomes, usually survival data. Right censoring occurs when instead of some outcome T, one can only observe $Y = \min(T, C)$ and $d = I(T \leq C)$, the indicator of failure. Left truncation occurs when Y is only observed subject to $Y > Z$. Typically, Z is the time at enrollement of subjects with a disease. Those who died before enrollment ($Y < Z$) cannot be observed, thus generating selection bias.

The formula must be of the form

- Surv(y,d) ~ x, with censored data;
- Surv(z,y,d) ~ x, with censored, truncated data;
- Surv(y) ~ x, is also allowed and denotes non-censored, non-truncated data.

In the above, x is a set of predictors, y is the response variable, z truncation times (z < y), and d the indicator of failure (1 = event, 0 = censored).

In flexPM, model fitting is implemented as follows. First, the response variable is pre-trasformed using a smoothed version of y = qlogis(rank(y)/(n + 1)). Second, parameter estimation is carried out on the transformed variable. Maximum likelihood estimators are computed via Newton-Raphson algorithm, using the following flexible distribution:

$$F_T(t \mid x) = \frac{1}{1 + e^{-\frac{u(t) - m(x)}{s(x)}}}.$$

In the above, $m(x)$ and $\log s(x)$ are modeled as specified by formula, while $u(\cdot)$ is a B-spline function built via spline.des (see bs). You can choose the degrees of freedom df and the degree of the spline basis. The model parameters are (a) the coefficients describing the effect of covariates x on $m(x)$ and $\log s(x)$, and (b) the coefficients of the B-spline basis that defines the unknown transformation $u(\cdot)$, on which suitable constraints are imposed to ensure monotonicity.

Value

An object of class "flexPM", which is a list with the following items:

converged	logical value indicating the convergence status of the algorithm.
n.it	the number of iterations.
n	the number of observations.
n.free.par	the number of free parameters in the model.
loglik	the values of the log-likelihood at convergence.
AIC, BIC	the Akaike and Bayesian information criterion.
mf	the used model frame.
call	the matched call.

The model parameters are returned as attributes of mf and are not user-level objects. The model is intended to be used for prediction and not for inference. The hessian matrix is not returned. The number of free parameters is df + 2*ncol(x) - 1, and not df + 2*ncol(x), because the scale of $u(\cdot)$ and that of $F_T(t \mid x)$ are exchangeable and thus one coefficient of $u(\cdot)$ is constrained to be 1.

The accessor functions summary, nobs, logLik, AIC, and BIC can be used to extract information from the model. The fit is only intended for prediction: use predict.flexPM.

Note

The model is fitted assuming that an unknown transformation of the response variable follows a Logistic distribution. The choice of the "kernel" distribution is only due to computational convenience and does not reflect any prior belief. Provided that $u(\cdot)$ is sufficiently flexible, asymmetric or multimodal distributions can be fitted. Pre-transforming the response variable (added in **flexPM 2.0**) removes the outliers, generates a more symmetric distribution, and frequently permits achieving a good fit using fewer knots for $u(\cdot)$.

This flexible parametric approach generally outoperforms fully nonparametric estimators like local Kaplan-Meier, at a cost of a relatively small bias.

Author(s)

Paolo Frumento <paolo.frumento@ki.se>

See Also

predict.flexPM

Examples

```
# Simulated data from a normal distribution

n <- 1000
x1 <- rnorm(n)
x2 <- runif(n)
```

```
# non-censored, non-truncated data

t <- rnorm(n, 2 + 3*x1, 1 + x2) # time variable
m1 <- flexPM(Surv(t) ~ x1 + x2)


# right-censored data

c <- rnorm(n,3,3) # censoring variable
y <- pmin(t,c)    # observed outcome
d <- (t <= c)     # 1 = observed, 0 = censored
m2 <- flexPM(Surv(y,d) ~ x1 + x2)


# right-censored, left-truncated data

z <- rnorm(n,-3,3) # truncating variable
w <- which(y > z)  # only observe if y > z
y <- y[w]; d <- d[w]; z <- z[w]; x1 <- x1[w]; x2 <- x2[w]
m3 <- flexPM(Surv(z,y,d) ~ x1 + x2)

#####################################################################

# m1, m2, m3 are not intended to be interpreted.
# Use predict() to obtain predictions.

# Note that the following are identical:
# summary(flexPM(Surv(y) ~ x1 + x2))
# summary(flexPM(Surv(y, rep(1,length(y))) ~ x1 + x2))
# summary(flexPM(Surv(rep(-Inf,length(y)), y, rep(1,length(y))) ~ x1 + x2))

#####################################################################

# Use the logLik, AIC and BIC for model selection
# (choice of df, inclusion/exclusion of covariates)

models <- list(
  flexPM(Surv(z,y,d) ~ x1 + x2, df = 1, degree = 1),
  flexPM(Surv(z,y,d) ~ x1 + x2, df = 3),
  flexPM(Surv(z,y,d) ~ x1 + x2 + I(x1^2) + I(x2^2), df = 1, degree = 1),
  flexPM(Surv(z,y,d) ~ x1 + x2 + I(x1^2) + I(x2^2), df = 3),
  flexPM(Surv(z,y,d) ~ x1 * x2, df = 5)
)

my_final_model <- models[[which.min(sapply(models, function(x) x$AIC))]]
summary(my_final_model)
```

`predict.flexPM` *Prediction from Fitted Flexible Parametric Models*

Description

Predicts the distribution function and simulates new data from a fitted model.

Usage

```
## S3 method for class 'flexPM'
predict(object, type = c("CDF", "QF", "sim"), newdata, p, ...)
```

Arguments

`object`	an object of class "`flexPM`".
`type`	the type of prediction (see 'Details').
`newdata`	an optional data frame in which to look for variables with which to predict. If omitted, the model frame of the object is used.
`p`	the order(s) of the quantile to be computed (for `type = "QF"`)
`...`	for future methods.

Details

- If `type = "CDF"` (the default), the fitted cumulative distribution function (CDF) and the corresponding probability density function (PDF) and survival function (SF) are returned.
- If `type = "QF"`, conditional quantiles of the specified order(s) are computed.
- If `type = "sim"`, data are simulated from the fitted model.

New data can be supplied: observe that for `type = "CDF"`, `newdata` must include the values of the response variable, and not just the covariates.

Value

- If `type = "CDF"`, a named data frame with variables `log.f` (the fitted log-PDF), `log.F` (the log-CDF) and `log.S` (the log-SF).
- If `type = "QF"`, a named data frame containing the fitted conditional quantiles of the specified order(s) in different columns.
- If `type = "sim"`, a vector of simulated data from the fitted model.

All types of prediction are computed at `newdata`, if supplied, or at the observed data, otherwise.

Author(s)

Paolo Frumento <paolo.frumento@ki.se>

See Also

`flexPM`

Examples

```
# Using simulated data

set.seed(1111); n <- 1000
x <- runif(n)
t <- rnorm(n, 1 + x, 1 + x)
model <- flexPM(Surv(t) ~ x + I(x^2))
# using polynomials (e.g. x^2) to achieve flexibility



# Prediction of the conditional cumulative distribution function (CDF)
# and the probability density function (PDF)

pred <- predict(model, type = "CDF") # predict the CDF and PDF

plot(pnorm(t, 1 + x, 1 + x), exp(pred$log.F))
abline(0,1, col = "green", lwd = 3) # true vs fitted CDF

plot(dnorm(t, 1 + x, 1 + x), exp(pred$log.f))
abline(0,1, col = "green", lwd = 3) # true vs fitted PDF



# Prediction of quantiles

predMe <- predict(model, type = "QF", p = 0.5) # predict the median
plot(x,t)
points(x, predMe$p0.5, col = "green") # fitted median
abline(1,1, col = "red", lwd = 3) # true median = 1 + x



# Simulate data from the fitted model

t.sim <- predict(model, type = "sim")
plot(quantile(t.sim, (1:9)/10), quantile(t, (1:9)/10)); abline(0,1)
# if the model is good, t and t.sim should have a similar distribution



######### Using new data ##############################

newdata <- data.frame(t = c(0,1,2), x = c(0.1,0.5,0.9))
# note that new 't' is only needed for type = "CDF"

predict(model, type = "CDF", newdata = newdata)
predict(model, type = "QF", newdata = newdata, p = c(0.25,0.5,0.75))
predict(model, type = "sim", newdata = newdata)
```

Index