Package ‘CommonSplines’

May 13, 2018

Title Regression Spline and Smoothing Spline

Version 1.0.0

Imports MASS

Date 2018-05-11

Authors Yuchen SHI <yuchenshinus@gmail.com>, Xiaozhou Yang <yang.xiaozhou@u.nus.edu>, Xingchen LIU <e0225109@u.nus.edu>

URL https://github.com/YuchenKid/CommonSplines

Description This is a R package that covers commonly seen nonparametric regression using spline-based methods. For regression spline, commonly seen basis functions are provided such as truncated power basis, natural cubic spline basis, and B-spline basis. For regularization, penalties on squared second-order derivative are provided, i.e., cubic smoothing spline.This package mainly refers to Chapter 5 of “Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning (Vol. 1, pp. 337-387). New York: Springer series in statistics".

Depends R (>= 3.3.2)

License Apache License 2.0

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Suggests knitr,
rmarkdown

VignetteBuilder knitr

R topics documented:

`bs_basis` *Generate an evaluated basis matrix for B-splines*

Description

This function generates B-spline basis. The B-splines are defined following the recursive formulas due to de Boor. Only univariate input can be used.

Usage

```
bs_basis(x, order, knots)
```

Arguments

`x`	Predictor variable vector.
`order`	The order of basis functions. order=degree+1
`knots`	The knots used to construct the B-splines, including innerknots, boundary knots and phantom knots. It can be generated by `bs_knots`.

Value

Basis matrix evaluated at each x value.

References

De Boor, C., De Boor, C., Mathématicien, E. U., De Boor, C., & De Boor, C. (1978). A Practical Guide to Splines (Vol. 27, p. 325). New York: Springer-Verlag.

Examples

```
x<-seq(0, 1, 0.001)
knots <- seq(0, 1, 0.1)
order<-4
knots<-bs_knots(x,real_knots=knots,order=order)

basis<-bs_basis(x,order,knots)
plot(x,rep(0,length(x)),type="l",ylim=c(0,1))
for (i in 1: (length(knots)-order)){
  lines(x,basis[,i],col=i)
}
```

bs_knots	*Add phantom knots for B-splines*

Description

Add phantom knots for B-splines

Usage

```
bs_knots(x, df = NULL, real_knots = NULL, q = FALSE, order)
```

Arguments

x	Predictor variable vector.
df	Degrees of freedom. One can supply df rather than knots.
real_knots	The innerknots and boundary knots that define the spline. The knots can all be innerknots. The knots provided can be quantiles of x or real values. More explanation of knots, df, q can be seen in generate_knots.
q	A boolean variable define whether knots provided are quantiles or real values. When q=TRUE, real_knots are quantiles of x. When q=FALSE, real_knots are real values of x. Default is FALSE.
order	The order of basis functions. order=degree+1

Value

The knots used to construct the B-splines, including innerknots, boundary knots and phantom knots.

bs_predict	*Prediction using regression spline with B-spline basis*

Description

This function provides prediction at value of interest using regression spline with B-spline basis. The B-splines are generated by bs_basis and trained by the bs_train. The return value of bs_train can be used as an argument of bs_predict

Usage

```
bs_predict(x_test, order = NULL, knots = NULL, beta = NULL,
  basis = NULL)
```

Arguments

x_test	The input values at which evaluations are required.
order	The order of basis functions. order=degree+1
knots	Breakpoints that define the spline. knots should be in terms of real-values of x and contain innner, boundary and phantom knots. It can be the return value of bs_knots.
beta	The coefficients of nonparametric regression.
basis	The return value of function bs_train. Instead of specify knots, order and beta,One can supply basis directly.

Value

The evaluated output at x_test.

See Also

bs_basis, bs_train, bs_knots.

Examples

```
x<-seq(0, 1, 0.001)
y <- x^3 * 3 - x^2 * 2 + x + exp(1)+rnorm(length(x),0,0.1)
plot(x,y)
knots <- seq(0.1, 0.9, 0.01)
order<-4
basis<-bs_train(x,y,order,knots)

x_test<-seq(0, 1, 0.01)
fit<-bs_predict(x_test,basis=basis)
plot(x_test,fit)
lines(x_test,x_test^3 * 3 - x_test^2 * 2 + x_test + exp(1),col="red")
```

---

bs_train *Train regression coefficients for B-splines.*

Description

Train regression coefficients for B-splines.

Usage

```
bs_train(x, y, order, real_knots = NULL, df = NULL, q = FALSE)
```

Arguments

x	The input vector of training dataset.
y	The output vector of training dataset.
order	The order of B-spline functions. The default is order=4 for cubic B-splines.
real_knots	The innerknots and boundary knots that define the spline. Phantom knots should not be included. Phantom knots will be generated by bs_knots The knots provided can be quantiles of x or real values. More explanation of knots, df, q can be seen in generate_knots.
df	Degrees of freedom. One can supply df rather than knots.
q	A boolean variable define whether knots provided are quantiles or real values. When q=TRUE, real_knots are quantiles of x. When q=FALSE, real_knots are real values of x. Default is FALSE.

Value

A list with the following components:

beta	The coefficients of nonparametric regression.
basis	The B-spline basis matrix of dimension c(length(x), df). df = length(innerknots) + order.
knots	The knots used to construct the B-splines, including innerknots, boundary knots and phantom knots
order	The order of basis functions. order=degree+1

References

Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning (Vol. 1, pp. 337-387). New York: Springer series in statistics. Chapter 5, Appendix.

See Also

bs_knots.

Examples

```
x<-seq(0, 1, 0.001)
y <- x^3 * 3 - x^2 * 2 + x + exp(1)+rnorm(length(x),0,0.1)
plot(x,y)
knots <- seq(0, 1, 0.1)
order<-4

basis<-bs_train(x,y,order,knots)
plot(x,rep(0,length(x)),type="l",ylim=c(0,1))
for (i in 1: (length(knots)+order)){
  lines(x,basis$basismatrix[,i],col=i)
}
```

---

cal_loo_cv_error *Calculte leave-one-out CV error for a linear smoother*

Description

Calculte leave-one-out CV error for a linear smoother

Usage

```
cal_loo_cv_error(y, f_hat, S)
```

Arguments

y	response variable values
f_hat	fitted response variable values
S	smoother matrix

Value

leave-one-out cross-validation error

css_predict — *Prediction using smoothing splines with squared 2nd derivative penalty*

Description

This function takes the coefficients trained by `css_train` and evaluates the output at x_test

Usage

```
css_predict(x_test, knots = NULL, beta = NULL, basis = NULL)
```

Arguments

`x_test`	The input values at which evaluations are required.
`knots`	Breakpoints that define the spline. `knots` should be in terms of real-values of x It can be the return value of `generate_knots`.
`beta`	The coefficients of nonparametric regression.
`basis`	The return value of function `css_train`. Instead of specify `knots` and `beta`,One can supply `basis` directly.

Value

The evaluated output at x_test.

Examples

```
x<-seq(0, 1, 0.0015)
y <- x^3 * 3 - x^2 * 2 + x + exp(1)+rnorm(length(x),0,0.1)
plot(x,y)
lambda<-0.001
basis<-css_train(x,y,lambda)

x_test<-seq(0, 1, 0.1)
fit<-css_predict(x_test=x_test,basis=basis)

plot(x_test,fit)
lines(x_test,x_test^3 * 3 - x_test^2 * 2 + x_test + exp(1),col="red")
```

css_train — *Train a smoothing spline with squared 2nd derivative penalty using natural cubic splines*

Description

This function trains a smoothing spline with squared 2nd derivative penalty. It has an explicit, finite-dimensional, unique minimizer which is a natural cubic spline.

Usage

```
css_train(x, y, lambda)
```

Arguments

x	The input vector of training dataset.
y	The output vector of training dataset.
lambda	A fixed smoothing parameter.

Value

A list with the following components:

beta	The coefficients of natural splines.
S	The smoother matrix.
knots	The knots used to construct the B-splines, including innerknots, boundary knots and phantom knots

References

Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning (Vol. 1, pp. 337-387). New York: Springer series in statistics. Chapter 5.4.

Examples

```
x<-seq(0, 1, 0.001)
y <- x^3 * 3 - x^2 * 2 + x + exp(1)+rnorm(length(x),0,0.1)
plot(x,y)
lambda<-0.001

basis<-css_train(x,y,lambda)
cat("the knots chosen are: ",basis$knots)
```

---

generate_knots *Generate knots when real values are not specified.*

Description

Generate knots when real values are not specified.

Usage

```
generate_knots(x_train, df = NULL, knots = NULL, q = FALSE)
```

Arguments

`x_train`	The input vector of training dataset.
`df`	Degrees of freedom. One can supply df rather than knots; `generate_knots` then chooses (df + 1) knots at uniform quantiles of x. The default, df = 4, sets 5 knots with 3 inner knots at uniform quantiles of x.
`knots`	Breakpoints that define the spline, in terms of quantiles or real valus of x. The default is five knots at uniform quantiles c(0, .25, .5, .75, 1). Typical values are the mean or median for one knot, quantiles for more knots.
`q`	A boolean variable define whether `knots` provided are quantiles or real values. When `q`=TRUE, `knots` provided are quantiles of x. When `q`=FALSE, `knots` provided are real values of x.

Value

A vector of knots in terms of real values of x.

`ncs_basis` *Generate an evaluated basis matrix for natural cubic splines*

Description

Generate an evaluated basis matrix for natural cubic splines

Usage

```
ncs_basis(x, knots)
```

Arguments

`x`	Predictor variable vector.
`knots`	Knots location in terms of real values of x.

Value

Basis matrix evaluated at each x value.

Examples

```
x<-seq(0, 1, 0.001)
knots <- seq(0, 1, 0.1)

basis<-ncs_basis(x,knots)
plot(x,rep(0,length(x)),type="l",ylim=c(0,1))
for (i in 1: (length(knots))){
  lines(x,basis[,i],col=i)
}
```

`ncs_predict` *Prediction using regression by natural cubic splines.*

Description

Prediction using regression by natural cubic splines.

Usage

```
ncs_predict(x_test, knots = NULL, beta = NULL, basis = NULL)
```

Arguments

`x_test`	The input values at which evaluations are required.
`knots`	Breakpoints that define the spline. `knots` should be in terms of real-values of x It can be the return value of `generate_knots`.
`beta`	The coefficients of nonparametric regression.
`basis`	The return value of function `ncs_train`. Instead of specify `knots` and `beta`,One can supply `basis` directly.

Value

`y_pred`	A vector of dimension length(x), the prediction vector evaluated at x_test values.

`ncs_train` *Train regression coefficients for natural cubic splines.*

Description

In the least square fitting of nonparametric egression coefficients, Moore-Penrose generalized inverse (ginv{MASS}) is used to aviod computational problems.

Usage

```
ncs_train(x_train, y_train, df = NULL, knots = NULL, q = FALSE)
```

Arguments

`x_train`	The input vector of training dataset.
`y_train`	The output vector of training dataset.
`df`	Degrees of freedom. One can supply df rather than knots; `ncs_train` then chooses (df + 1) knots at uniform quantiles of x. The default, df = 4, sets 5 knots with 3 inner knots at uniform quantiles of x.
`knots`	Breakpoints that define the spline, in terms of quantiles of x or real values of x. The default is five knots at uniform quantiles c(0, .25, .5, .75, 1). Typical values are the mean or median for one knot, quantiles for more knots.
`q`	A boolean variable indicating whether `knots` provided are quantiles or real values. When q=TRUE, `knots` provided are quantiles of x. When q=FALSE, `knots` provided are real values of x. Default is FALSE.

Value

A list of following components:

nknots	Number of knots.
knots	A vector of knot locations.
N	Basis matrix evaluated at each x value.
betas	Least sqaure fit parameters.

References

Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning (Vol. 1, pp. 337-387). New York: Springer series in statistics. Chapter 5.2.1.

Venables, W. N. and Ripley, B. D. (1999) Modern Applied Statistics with S-PLUS. Third Edition. Springer. p.100.

Examples

```
x_train <- seq(1, 10, 0.1)
y_train <- cos(x_train)^3 * 3 - sin(x_train)^2 * 2 + x_train + exp(1)+rnorm(length(x_train),0,1)
plot(x_train,y_train)
x_test <- seq(1, 10, 0.1)
df <- 10
train_result <- ncs_train(x_train, y_train, df)
print(train_result$beta)
print(train_result$N[1:5,1:5])
```

---

np_reg *Nonparametric regression using spline-based methods*

Description

This function provides regression using spline-based methods. It finishes both the training and predicting procedure. Only univariate input can be used.

Usage

```
np_reg(x_train, y_train, x_test, func = "bs", order = 4, df = NULL,
  knots = NULL, lambda = 0.001, q = FALSE)
```

Arguments

x_train	The input vector of training dataset.
y_train	The output vector of training dataset.
x_test	The input values at which evaluations are required.
func	The name of regression functions. It can be "pbs" for power basis splines, "ncs" for natural cubic splines, "css" for cubic smoothing splines, "bs" for B-splines. Default is "bs".
order	The order that defines the spline. Default is 4 of a cubic order.
df	Degrees of freedom. One can supply df rather than knots.

knots	The innerknots and boundary knots that define the spline. The knots provided can be quantiles of x or real values. More explanation of `knots`, `df`, `q` can be seen in `generate_knots`.
lambda	The smoothing parameter for css. Default is 0.001.
q	A boolean variable indicating whether `knots` provided are quantiles or real values. When q=TRUE, `knots` provided are quantiles of x. When q=FALSE, `knots` provided are real values of x. Default is FALSE.

Value

y_pred	A vector of dimension `length(x)`, the prediction vector evaluated at x_test values.

References

Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning (Vol. 1, pp. 337-387). New York: Springer series in statistics. Chapter 5.

See Also

`generate_knots`.

Examples

```
x_train <- seq(1, 10, 0.1)
y_train <- cos(x_train)^3 * 3 - sin(x_train)^2 * 2 + x_train + exp(1)+rnorm(length(x_train),0,1)
plot(x_train,y_train)
title('Comparison of Different Degrees of Freedom')
x_test <- seq(1, 10, 0.1)
lines(x_test,cos(x_test)^3 * 3 - sin(x_test)^2 * 2 + x_test + exp(1),col="red")

df <- 2
y_pred <- np_reg(x_train, y_train, x_test,func="ncs", df=df)
lines(x_test,y_pred, col='blue')
df <- 4
y_pred <- np_reg(x_train, y_train, x_test,func="ncs", df=df)
lines(x_test,y_pred, col='green')
df <- 10
y_pred <- np_reg(x_train, y_train, x_test,func="ncs", df=df)
lines(x_test,y_pred, col='black')
legends <- c("Actual", "Prediction: 2 df", "Prediction: 4 df", "Prediction: 10 df")
legend('topleft', legend=legends, col=c('red', 'blue', 'green', 'black'), lty=1, cex=0.8)
```

`pbs_basis` *Evaluate basis functions at each x and return the evaluated basis matrix N*

Description

Evaluate basis functions at each x and return the evaluated basis matrix N

Usage

```
pbs_basis(x, order, knots)
```

Arguments

`x`	The input vector of training dataset.
`order`	The order that defines the power basis spline.
`knots`	The innerknots and boundary knots that define the spline. The knots should be real values of x. The knots can be generated by `generate_knots`.

Value

Basis matrix evaluated at each x value.

See Also

`generate_knots`.

Examples

```
x<-seq(0, 1, 0.001)
knots <- seq(0, 1, 0.1)
order<-4

basis<-pbs_basis(x,order,knots)
plot(x,rep(0,length(x)),type="l",ylim=c(0,1))
for (i in 1: (length(knots)+order)){
  lines(x,basis[,i],col=i)
}
```

`pbs_predict` *Prediction using regression spline with trancated power basis*

Description

This function provides prediction at value of interest using regression spline with truncated power basis. The truncated power basis are generated by `pbs_basis` and trained by the `pbs_train`. The return value of `pbs_train` can be used as an argument of `pbs_predict`

Usage

```
pbs_predict(x_test, order = NULL, knots = NULL, beta = NULL,
  basis = NULL)
```

Arguments

`x_test`	The input values at which evaluations are required.
`order`	The order of basis functions. order=degree+1
`knots`	Breakpoints that define the spline, in terms of real values of input. It can be the return value of `generate_knots`.
`beta`	The coefficients of nonparametric regression.
`basis`	The return value of function `pbs_train`. Instead of specify `knots`, `order` and `beta`,One can supply `basis` directly.

Value

The evaluated output at x_test.

See Also

pbs_basis, pbs_train, generate_knots.

Examples

```
n <- 100
t <- seq(0,2*pi,length.out = 100)
a <- 3
b <- 2
c.unif <- runif(n)
amp <- 2
set.seed(1)
y1 <- a*sin(b*t)+c.unif*amp # uniform error
knots <- c(min(t),2*pi*c(1/4,2/4,3/4),max(t))
order <- 4
basis <- pbs_train(t,y1,order,knots=knots)
fit<-pbs_predict(t,basis=basis)
y.hat <- fit
plot(t, y1, t="l")
lines(t, y.hat, col=2)
```

---

pbs_train *Regression using Power Basis spline*

Description

This function provides regressions using Power Basis splines. The basis are defined as 1,x,x^2,...,x^m,(x-k1)^(m-1)+,(x-k2)^(m-1)+,...,(x-kn)^(m-1)+ where m is the order, k1, k2 and kn are n knots, '+' denotes the positive part. Only univariate input can be used.

Usage

```
pbs_train(x, y, order, df = NULL, knots = NULL, q = FALSE)
```

Arguments

x	The input vector of training dataset.
y	The output vector of training dataset.
order	The order that defines the spline.
df	Degrees of freedom. One can supply df rather than knots.
knots	The innerknots and boundary knots that define the spline. The knots provided can be quantiles of x or real values of x. More explanation of knots, df, q can be seen in generate_knots.
q	A boolean variable define whether knots provided are quantiles or real values. When q=TRUE, knots provided are quantiles of x. When q=FALSE, knots provided are real values of x. Default is FALSE.
x_test	The input values at which evaluations are required.

Value

A list with the following components:

beta	The coefficients of nonparametric regression.
basis	The spline basis matrix of dimension c(length(x), length(knots)+order)
knots	The knots used to construct the power basis splines
order	The order of basis functions. order=degree+1

References

Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning (Vol. 1, pp. 337-387). New York: Springer series in statistics. Chapter 5.2.1.

See Also

generate_knots.

Examples

```
n <- 100
t <- seq(0,2*pi,length.out = 100)
a <- 3
b <- 2
c.unif <- runif(n)
amp <- 2
set.seed(1)
y1 <- a*sin(b*t)+c.unif*amp # uniform error
knots <- c(min(t),2*pi*c(1/4,2/4,3/4),max(t))
order <- 4
basis <- pbs_train(t,y1,order,knots=knots)
cat("trained coeffecients for every spline are",basis$beta)
```

---

place_knots *Find evenly spaced knots by quantile*

Description

Knots found include boundary knots at 0th and 100th quantile.

Usage

```
place_knots(nknots, x)
```

Arguments

nknots	Number of knots to be located.
x	Data vector on which knots are placed.

Value

A named vector with knot quantiles and values.

`sel_smoothing_para` *Select smoothing parameter for smoothing splines based on leave-one-out CV error*

Description

Select smoothing parameter for smoothing splines based on leave-one-out CV error

Usage

```
sel_smoothing_para(x, y, cv_lambda)
```

Arguments

x	predictor variable.
y	response variable.
cv_lambda	vector of candidate lambda values, must be between 0 and 1.

Value

lamdba value that minimizes leave-one-out CV error.

Examples

```
set.seed(1)
x_train <- seq(1, 10, 0.1)
y_train <- cos(x_train)^3 * 3 - sin(x_train)^2 * 2 + x_train + exp(1)+rnorm(length(x_train),0,1)
plot(x_train,y_train)
x_test <- seq(1, 10, 0.1)
lines(x_test,cos(x_test)^3 * 3 - sin(x_test)^2 * 2 + x_test + exp(1),col="red")

cv_lambda <- seq(0.0001,0.002,0.0001)
results <- sel_smoothing_para(x_train, y_train, cv_lambda)
plot(results$df$cv_lambda, results$df$error, type="o")
title('LOO CV Error of Different Lambdas')
abline(v = results$best, col='red', lty=2)
legends <- c("LOO CV Error", "Best Lambda")
legend('topleft', legend=legends, col=c('black', 'red'), lty=c(1,2), cex=0.8)
```

Index