Package 'cbass'

July 6, 2023

Version 0.1

Date 2023-06-15

Title Classification -- Bayesian Adaptive Smoothing Splines

Description

Fit multiclass Classification version of Bayesian Adaptive Smoothing Splines (CBASS) to data using reversible jump MCMC. The multiclass classification problem consists of a response variable that takes on unordered categorical values with at least three levels, and a set of inputs for each response variable. The CBASS model consists of a latent multivariate probit formulation, and the means of the latent Gaussian random variables are specified using adaptive regression splines. The MCMC alternates updates of the latent Gaussian variables and the spline parameters. All the spline parameters (variables, signs, knots, number of interactions), including the number of basis functions used to model each latent mean, are inferred. Functions are provided to process inputs, initialize the chain, run the chain, and make predictions. Predictions are made on a probabilistic basis, where, for a given input, the probabilities of each categorical value are produced. See Marrs and Francom (2023) ``Multiclass classification using Bayesian multivariate adaptive regression splines'' Under review.

RoxygenNote 7.2.3

License GPL-3

NeedsCompilation no

Author Frank Marrs [aut, cre] (<https://orcid.org/0000-0003-3445-9170>), Devin Francom [aut] (<https://orcid.org/0000-0002-6009-4438>)

Maintainer Frank Marrs <fmarrs3@lanl.gov>

Repository CRAN

Date/Publication 2023-07-06 19:10:05 UTC

R topics documented:

augment.X *Augment X for missing data approach for MNAR*

Description

Augment X for missing data approach for MNAR

Usage

```
augment.X(X)
```

Arguments

X	matrix of covariates, including some missing values (NAs)

Value

Matrix same size as X, with augmented columns and zeros in the missing spots

Examples

```
set.seed(1)
n <- 100
X <- matrix(runif(n*2, 0, 1), ncol=2)
X[sample(1:length(X), round(.1*length(X)))] <- NA
X.new <- augment.X(X)
sum(is.na(X.new))
```

fit.cbass *Fit CBASS model using reversible jump MCMC*

Description

Fit CBASS model using reversible jump MCMC

Usage

```
fit.cbass(
  X,
  y,
  max.int = 3,
  max.basis = 10 * ncol(X),
  tau2 = 10,
  nmcmc = 10000,
  nburn = round(nmcmc/2),
  nthin = 10,
```

```
  h1 = 4,
  h2 = 20 * (length(unique(y)) - 1)/nrow(X),
  p.int.prior = 1/(1:max.int),
  verbose = FALSE,
  print.interval = round(nmcmc/100),
  init1 = FALSE,
  ordinal = NULL,
  writeout = FALSE,
  writedir = tempdir(),
  mod = NULL,
  restart = FALSE
)
```

Arguments

X	n by p matrix of inputs on unit invteral
y	n-length factor factor of categories
max.int	maximum number of interactions, default 3
max.basis	maximum number of basis functions for each latent mean function, default ncol(X)*10
tau2	prior variance of basis regression coefficients, default 10
nmcmc	number of MCMC samples, default 1e4
nburn	number of MCMC samples to burn, default nmcmc/2
nthin	number of samples by which to thin, default 10
h1	first parameter for Gamma hyperprior on tau2, default 4
h2	second parameter for Gamma hyperprior on tau2, default 20(d-1)/n
p.int.prior	prior for number of interactions, default 1/(1:max.int)
verbose	should progress be printed out? default false
print.interval	how often should progress be printed out? default every 1%
init1	should model be initialized with single interaction model? default FALSE
ordinal	indicator of ordinal predictors (non-categorical), usually computed automatically
writeout	should samples be written out to text file? default FALSE
writedir	where should text files be written? default tempdir()
mod	initial / previous model fit, default NULL
restart	should initial input model be used for starting chain? default FALSE

Value

A list of CBASS model parameters. LIST THEM.

Examples

```
set.seed(1)
n <- 100;  d <- 3
X <- matrix(runif(n*2, 0, 1), ncol=2)
mu <- scale(X)
bound <- qnorm(1/d^(1/(d-1)))
mu <- cbind(bound, mu)
z <- mu
z[,-1] <- rnorm(length(mu[,-1]), mu[,-1], 1)
y <- apply(z, 1, which.max)
mod <- fit.cbass(X, y, max.int=1, max.basis=10, nmcmc=1e3, nburn=500, nthin=10)
pred.chain <- pred.cbass(mod, X)
mu.hat <- apply(pred.chain, 2:3, mean)
mean(abs(mu - mu.hat))
plot(c(mu), c(mu.hat))
```

---

p.mu *Predict vector of probabilities from vector of latent means*

Description

Predict vector of probabilities from vector of latent means

Usage

```
p.mu(mu, d = NULL, bound = NULL, npts = 100, rel.tol = 1e-04)
```

Arguments

mu	d-length vector of latent means
d	input to avoid computing length of mu
bound	input of mu[1] to avoid computation
npts	number of integration points, default 100
rel.tol	number of integration points, default 1e-4

Value

A d-length numeric vector of probabilities given input latent means

Examples

```
set.seed(1)
mu <- rnorm(5)
p.mu(mu)
```

`pred.cbass` *Generate chain of latent normal random variables for a given X, for values saved in 'mod'*

Description

Generate chain of latent normal random variables for a given X, for values saved in 'mod'

Usage

```
pred.cbass(mod, X, nburn = 0, nsub = NULL)
```

Arguments

`mod`	CBASS model list
`X`	matrix of covariates of same size / makeup as that used to create mod. If matrix not scaled to the unit interval, then it will be
`nburn`	Number of samples to burn from the chain in mod, default 0
`nsub`	Number of samples to subset to, default to those stored in mod

Value

An array of latent variables, nsub by nrow(X) by d

Examples

```
set.seed(1)
n <- 100;  d <- 3
X <- matrix(runif(n*2, 0, 1), ncol=2)
mu <- scale(X)
bound <- qnorm(1/d^(1/(d-1)))
mu <- cbind(bound, mu)
z <- mu
z[,-1] <- rnorm(length(mu[,-1]), mu[,-1], 1)
y <- apply(z, 1, which.max)
mod <- fit.cbass(X, y, max.int=1, max.basis=10, nmcmc=1e3, nburn=500, nthin=10)
pred.chain <- pred.cbass(mod, X)
mu.hat <- apply(pred.chain, 2:3, mean)
round(p.mu(mu.hat[1,]), 3)
```

`sample.z` *Draw samples of independent normals (matrix) given previous sample, and maximal values*

Description

Draw samples of independent normals (matrix) given previous sample, and maximal values

Usage

```
sample.z(mu, y, z)
```

Arguments

`mu`	n by d matrix of latent means
`y`	n-length vector of maximal indices
`z`	n by d matrix of latent random variables

Value

A new sample of n by d matrix of latent random variables

Examples

```
set.seed(1)
n <- 100;  d <- 3
mu <- matrix(rnorm(n*d), n, d)
bound <- qnorm(1/d^(1/(d-1)))
mu[,1] <- bound
z <- mu
z[,-1] <- rnorm(length(mu[,-1]), mu[,-1], 1)
y <- apply(z, 1, which.max)
z.new <- sample.z(mu, y, z)
all(apply(z.new, 1, which.max) == y)
```

Index