# Package ‘calibrateBinary’

July 20, 2018

**Type** Package

**Title** Calibration for Computer Experiments with Binary Responses

**Version** 0.1

**Date** 2018-07-16

**Author** Chih-Li Sung

**Maintainer** Chih-Li Sung <iamdfchile@gmail.com>

**Description** Performs the calibration procedure proposed by Sung et al. (2018+) <arXiv:1806.01453>. This calibration method is particularly useful when the outputs of both computer and physical experiments are binary and the estimation for the calibration parameters is of interest.

**License** GPL-2 | GPL-3

**RoxygenNote** 5.0.1

**Depends** R (>= 2.14.1)

**Imports** GPfit, gelnet, kernlab, randtoolbox

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-07-20 15:00:07 UTC

## R topics documented:

calibrateBinary *Calibration for Binary Outputs*

## Description

The function performs the L2 calibration method for binary outputs.

## Usage

```
calibrateBinary(Xp, yp, Xs1, Xs2, ys, K = 5, lambda = seq(0.001, 0.1,
  0.005), kernel = c("matern", "exponential")[1], nu = 1.5, power = 1.95,
  rho = seq(0.05, 0.5, 0.05), sigma = seq(100, 20, -1), lower, upper,
  verbose = TRUE)
```

## Arguments

| | |
|---|---|
| Xp | a design matrix with dimension np by d. |
| yp | a response vector with length np. The values in the vector are 0 or 1. |
| Xs1 | a design matrix with dimension ns by d. These columns should one-by-one correspond to the columns of Xp. |
| Xs2 | a design matrix with dimension ns by q. |
| ys | a response vector with length ns. The values in the vector are 0 or 1. |
| K | a positive integer specifying the number of folds for fitting kernel logistic regression and generalized Gaussian process. The default is 5. |
| lambda | a vector specifying lambda values at which CV curve will be computed for fitting kernel logistic regression. See cv.KLR. |
| kernel | input for fitting kernel logistic regression. See KLR. |
| nu | input for fitting kernel logistic regression. See KLR. |
| power | input for fitting kernel logistic regression. See KLR. |
| rho | rho value at which CV curve will be computed for fitting kernel logistic regression. See KLR. |
| sigma | a vector specifying values of the tuning parameter $\sigma$ at which CV curve will be computed for fitting generalized Gaussian process. See Details. |
| lower | a vector of size p+q specifying lower bounds of the input space for rbind(Xp,Xs1) and Xs2. |
| upper | a vector of size p+q specifying upper bounds of the input space for rbind(Xp,Xs1) and Xs2. |
| verbose | logical. If TRUE, additional diagnostics are printed. The default is TRUE. |

## Details

The function performs the L2 calibration method for computer experiments with binary outputs. The input and ouput of physical data are assigned to `Xp` and `yp`, and the input and output of computer data are assigned to `cbind(Xs1,Xs2)` and `ys`. Note here we separate the input of computer data by `Xs1` and `Xs2`, where `Xs1` is the shared input with `Xp` and `Xs2` is the calibration input. The idea of L2 calibration is to find the calibration parameter that minimizes the discrepancy measured by the L2 distance between the underlying probability functions in the physical and computer data. That is,

$$\hat{\theta} = \arg\min_{\theta} \|\hat{\eta}(\cdot) - \hat{p}(\cdot, \theta)\|_{L_2(\Omega)},$$

where $\hat{\eta}(x)$ is the fitted probability function for physical data, and $\hat{p}(x, \theta)$ is the fitted probability function for computer data. In this L2 calibration framework, $\hat{\eta}(x)$ is fitted by the kernel logistic regression using the input `Xp` and the output `yp`. The tuning parameter $\lambda$ for the kernel logistic regression can be chosen by k-fold cross-validation, where k is assigned by `K`. The choices of the tuning parameter are given by the vector `lambda`. The kernel function for the kernel logistic regression can be given by `kernel`, where Matern kernel or power exponential kernel can be chosen. The arguments `power`, `nu`, `rho` are the tuning parameters in the kernel functions. See `KLR`. For computer data, the probability function $\hat{p}(x, \theta)$ is fitted by the Bayesian Gaussian process in Williams and Barber (1998) using the input `cbind(Xs1,Xs2)` and the output `ys`, where the Gaussian correlation function,

$$R_{\sigma}(\mathbf{x}_i, \mathbf{x}_j) = \exp\{-\sum_{l=1}^{d} \sigma(x_{il} - x_{jl})^2\},$$

is used here. The vector `sigma` is the choices of the tuning parameter $\sigma$, and it will be chosen by k-fold cross-validation. More details can be seen in Sung et al. (unpublished). The arguments `lower` and `upper` are lower and upper bounds of the input space, which will be used in scaling the inputs and optimization for $\theta$. If they are not given, the default is the range of each column of `rbind(Xp,Xs1)`, and `Xs2`.

## Value

a matrix with number of columns `q+1`. The first `q` columns are the local (the first row is the global) minimal solutions which are the potential estimates of calibration parameters, and the `(q+1)`-th column is the corresponding L2 distance.

## Author(s)

Chih-Li Sung <iamdfchile@gmail.com>

## See Also

`KLR` for performing a kernel logistic regression with given `lambda` and `rho`. `cv.KLR` for performing cross-validation to estimate the tuning parameters.

## Examples

```
library(calibrateBinary)

set.seed(1)
```

```
#####   data from physical experiment   #####
np <- 10
xp <- seq(0,1,length.out = np)
eta_fun <- function(x) exp(exp(-0.5*x)*cos(3.5*pi*x)-1) # true probability function
eta_x <- eta_fun(xp)
yp <- rep(0,np)
for(i in 1:np) yp[i] <- rbinom(1,1, eta_x[i])

#####   data from computer experiment   #####
ns <- 20
xs <- matrix(runif(ns*2), ncol=2)  # the first column corresponds to the column of xp
p_xtheta <- function(x,theta) {
     # true probability function
     exp(exp(-0.5*x)*cos(3.5*pi*x)-1) - abs(theta-0.3) *exp(-0.5*x)*cos(3.5*pi*x)
}
ys <- rep(0,ns)
for(i in 1:ns) ys[i] <- rbinom(1,1, p_xtheta(xs[i,1],xs[i,2]))

#####    check the true parameter    #####
curve(eta_fun, lwd=2, lty=2, from=0, to=1)
curve(p_xtheta(x,0.3), add=TRUE, col=4)   # true value = 0.3: L2 dist = 0
curve(p_xtheta(x,0.9), add=TRUE, col=3)   # other value

##### calibration: true parameter is 0.3 #####

calibrate.result <- calibrateBinary(xp, yp, xs[,1], xs[,2], ys)
print(calibrate.result)
```

---

`cv.KLR` *K-fold cross-validation for Kernel Logistic Regression*

---

## Description

The function performs k-fold cross validation for kernel logistic regression to estimate tuning parameters.

## Usage

```
cv.KLR(X, y, K = 5, lambda = seq(0.001, 0.2, 0.005), kernel = c("matern",
  "exponential")[1], nu = 1.5, power = 1.95, rho = seq(0.05, 0.5, 0.05))
```

## Arguments

| | |
|---|---|
| `X` | input for `KLR`. |
| `y` | input for `KLR`. |
| `K` | a positive integer specifying the number of folds. The default is 5. |
| `lambda` | a vector specifying lambda values at which CV curve will be computed. |

| | |
|---|---|
| kernel | input for KLR. |
| nu | input for KLR. |
| power | input for KLR. |
| rho | rho value at which CV curve will be computed. |

## Details

This function performs the k-fold cross-valibration for a kernel logistic regression. The CV curve is computed at the values of the tuning parameters assigned by `lambda` and `rho`. The number of fold is given by `K`.

## Value

| | |
|---|---|
| lambda | value of `lambda` that gives minimum CV error. |
| rho | value of `rho` that gives minimum CV error. |

## Author(s)

Chih-Li Sung <iamdfchile@gmail.com>

## See Also

KLR for performing a kernel logistic regression with given `lambda` and `rho`.

## Examples

```
library(calibrateBinary)

set.seed(1)
np <- 10
xp <- seq(0,1,length.out = np)
eta_fun <- function(x) exp(exp(-0.5*x)*cos(3.5*pi*x)-1) # true probability function
eta_x <- eta_fun(xp)
yp <- rep(0,np)
for(i in 1:np) yp[i] <- rbinom(1,1, eta_x[i])

x.test <- seq(0,1,0.001)
etahat <- KLR(xp,yp,x.test)

plot(xp,yp)
curve(eta_fun, col = "blue", lty = 2, add = TRUE)
lines(x.test, etahat, col = 2)

#####   cross-validation with K=5    #####
##### to determine the parameter rho #####

cv.out <- cv.KLR(xp,yp,K=5)
print(cv.out)

etahat.cv <- KLR(xp,yp,x.test,lambda=cv.out$lambda,rho=cv.out$rho)
```

```
plot(xp,yp)
curve(eta_fun, col = "blue", lty = 2, add = TRUE)
lines(x.test, etahat, col = 2)
lines(x.test, etahat.cv, col = 3)
```

KLR *Kernel Logistic Regression*

## Description

The function performs a kernel logistic regression for binary outputs.

## Usage

```
KLR(X, y, xnew, lambda = 0.01, kernel = c("matern", "exponential")[1],
  nu = 1.5, power = 1.95, rho = 0.1)
```

## Arguments

| | |
|---|---|
| X | a design matrix with dimension n by d. |
| y | a response vector with length n. The values in the vector are 0 or 1. |
| xnew | a testing matrix with dimension n_new by d in which each row corresponds to a predictive location. |
| lambda | a positive value specifing the tuning parameter for KLR. The default is 0.01. |
| kernel | "matern" or "exponential" which specifies the matern kernel or power exponential kernel. The default is "matern". |
| nu | a positive value specifying the order of matern kernel if kernel == "matern". The default is 1.5 if matern kernel is chosen. |
| power | a positive value (between 1.0 and 2.0) specifying the power of power exponential kernel if kernel == "exponential". The default is 1.95 if power exponential kernel is chosen. |
| rho | a positive value specifying the scale parameter of matern and power exponential kernels. The default is 0.1. |

## Details

This function performs a kernel logistic regression, where the kernel can be assigned to Matern kernel or power exponential kernel by the argument kernel. The arguments power and rho are the tuning parameters in the power exponential kernel function, and nu and rho are the tuning parameters in the Matern kernel function. The power exponential kernel has the form

$$K_{ij} = \exp(-\frac{\sum_k |x_{ik} - x_{jk}|^{power}}{rho}),$$

and the Matern kernel has the form

$$K_{ij} = \prod_k \frac{1}{\Gamma(nu)2^{nu-1}} (2\sqrt{nu}\frac{|x_{ik} - x_{jk}|}{rho})^{nu} \kappa(2\sqrt{nu}\frac{|x_{ik} - x_{jk}|}{rho}).$$

The argument `lambda` is the tuning parameter for the function smoothness.

## Value

Predictive probabilities at given locations `xnew`.

## Author(s)

Chih-Li Sung <iamdfchile@gmail.com>

## References

Zhu, J. and Hastie, T. (2005). Kernel logistic regression and the import vector machine. Journal of Computational and Graphical Statistics, 14(1), 185-205.

## See Also

`cv.KLR` for performing cross-validation to choose the tuning parameters.

## Examples

```
library(calibrateBinary)

set.seed(1)
np <- 10
xp <- seq(0,1,length.out = np)
eta_fun <- function(x) exp(exp(-0.5*x)*cos(3.5*pi*x)-1) # true probability function
eta_x <- eta_fun(xp)
yp <- rep(0,np)
for(i in 1:np) yp[i] <- rbinom(1,1, eta_x[i])

x.test <- seq(0,1,0.001)
etahat <- KLR(xp,yp,x.test)

plot(xp,yp)
curve(eta_fun, col = "blue", lty = 2, add = TRUE)
lines(x.test, etahat, col = 2)

#####   cross-validation with K=5    #####
##### to determine the parameter rho #####

cv.out <- cv.KLR(xp,yp,K=5)
print(cv.out)

etahat.cv <- KLR(xp,yp,x.test,lambda=cv.out$lambda,rho=cv.out$rho)

plot(xp,yp)
```

```
curve(eta_fun, col = "blue", lty = 2, add = TRUE)
lines(x.test, etahat, col = 2)
lines(x.test, etahat.cv, col = 3)
```

# Index