Package ‘WLogit’

January 20, 2025

Type Package

Title Variable Selection in High-Dimensional Logistic Regression Models using a Whitening Approach

Version 2.1

Date 2023-07-17

Author Wencan Zhu

Maintainer Wencan Zhu <wencan.zhu@yahoo.com>

Description It proposes a novel variable selection approach in classification problem that takes into account the correlations that may exist between the predictors of the design matrix in a high-dimensional logistic model. Our approach consists in rewriting the initial high-dimensional logistic model to remove the correlation between the predictors and in applying the generalized Lasso criterion.

License GPL-2

Imports cvCovEst, genlasso, tibble, MASS, ggplot2, Matrix, glmnet, corpcor

VignetteBuilder knitr

Suggests knitr

Depends R (>= 3.5.0)

NeedsCompilation no

Repository CRAN

Date/Publication 2023-07-17 07:10:06 UTC

Contents

`WLogit-package` *Variable Selection in High-Dimensional Logistic Regression Models using a Whitening Approach*

Description

It proposes a novel variable selection approach in classification problem that takes into account the correlations that may exist between the predictors of the design matrix in a high-dimensional logistic model. Our approach consists in rewriting the initial high-dimensional logistic model to remove the correlation between the predictors and in applying the generalized Lasso criterion.

Details

The DESCRIPTION file:

Package:	WLogit
Type:	Package
Title:	Variable Selection in High-Dimensional Logistic Regression Models using a Whitening Approach
Version:	2.1
Date:	2023-07-17
Author:	Wencan Zhu
Maintainer:	Wencan Zhu <wencan.zhu@yahoo.com>
Description:	It proposes a novel variable selection approach in classification problem that takes into account the corre
License:	GPL-2
Imports:	cvCovEst, genlasso, tibble, MASS, ggplot2, Matrix, glmnet, corpcor
VignetteBuilder:	knitr
Suggests:	knitr
Depends:	R (>= 3.5.0)
NeedsCompilation:	no
Packaged:	2023-07-17 07:06:43 UTC; mmip

Index of help topics:

```
CalculPx                Calculate the class-conditional probabilities.
CalculWeight            Calculate the weight
Refit_glm               Refit the logistic regression with chosen
                        variables
Thresholding            Thresholding on a vector
```

WLogit-package	Variable Selection in High-Dimensional Logistic Regression Models using a Whitening Approach
WhiteningLogit	Variable selection in high-dimensional logistic regression models using a whitening approach
WorkingResp	Calculate the working response
X	Example of a design matrix of a logistic model
beta	True coefficients in the esample.
test	WLogit output
top	Thresholding to zero of the smallest values
top_thresh	Thresholding to a given threshold of the smallest values
y	Example of a binary response variable of a logistic model.

Further information is available in the following vignettes:

Vignettes	WLogit package (source, pdf)

This package consists of functions: "WhiteningLogit", "CalculPx", "CalculWeight", "Refit_glm", "top", "top_thresh", "WorkingResp", and "Thresholding". For further information on how to use these functions, we refer the reader to the vignette of the package.

Author(s)

Wencan Zhu

Maintainer: Wencan Zhu <wencan.zhu@yahoo.com>

References

W. Zhu, C. Levy-Leduc, N. Ternes. "Variable selection in high-dimensional logistic regression models using a whitening approach". (2022)

beta *True coefficients in the esample.*

Description

True coefficients in the esample given in the vignette.

Usage

```
data("beta")
```

Format

The format is: num [1:500] 1 1 1 1 1 1 1 1 1 1 ...

Examples

```
data(beta)
plot(beta)
```

CalculPx *Calculate the class-conditional probabilities.*

Description

Calculate the probability for a repsonse to be 1 in the logistic regression model.

Usage

```
CalculPx(X, beta, intercept = 0)
```

Arguments

`X`	Design matrix of the logistic model considered.
`beta`	Vector of coefficients of the logistic model considered.
`intercept`	Whether there is the intercept

Value

`prob`	the probability for a repsonse to be 1

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

See Also

Please read https://hastie.su.domains/Papers/glmnet.pdf for more details

Examples

```
data(X)
data(beta)
CalculPx(X=X, beta=beta)


##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.
```

```
## The function is currently defined as
function (X, beta, intercept = 0)
{
    prob <- 1/(1 + exp(-(X %*% beta + intercept)))
    return(prob)
  }
```

`CalculWeight` *Calculate the weight*

Description

Calculate the weight in the penalized weighted- least-squares problem

Usage

```
CalculWeight(Px)
```

Arguments

`Px`	The vector of estimated probability for each response to be 1.

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

See Also

Please read `https://hastie.su.domains/Papers/glmnet.pdf` for more details

Examples

```
data(X)
data(beta)
px <- CalculPx(X=X, beta=beta)
CalculWeight(px)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (Px)
{
    return(Px * (1 - Px))
  }
```

Refit_glm *Refit the logistic regression with chosen variables*

Description

Refit the logistic regression with chosen variables.

Usage

```
Refit_glm(X, beta_pred, y)
```

Arguments

X	Design matrix of the logistic model considered.
beta_pred	Predicted coefficients to be refited.
y	Binary response

Value

beta_refit	The new estimated coefficients

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

Examples

```
data(X)
data(y)
data(beta)
Refit_glm(X=X, beta_pred=beta, y=y)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (X, beta_pred, y)
{
    X_temp <- X[, which(beta_pred != 0)]
    if (length(which(beta_pred != 0)) == 0) {
        coef_est <- beta_pred
    }
    else if (is.null(ncol(X_temp))) {
        mydata <- data.frame(Y = y, X_temp)
        colnames(mydata) <- c("Y", "X")
        formula <- paste0("Y~-1 +", paste0(colnames(mydata)[-which(colnames(mydata) ==
            "Y")], collapse = " + "))
        myform <- as.formula(formula)
        mod_lm <- glm(myform, data = mydata, family = "binomial")
```

```
      coef_est <- mod_lm$coefficients
  }
  else {
      mydata <- data.frame(Y = y, as.matrix(X_temp))
      formula <- paste0("Y~-1 +", paste0(colnames(mydata)[-which(colnames(mydata) ==
          "Y")], collapse = " + "))
      myform <- as.formula(formula)
      if (length(which(beta_pred != 0)) >= length(y)) {
          mod_ridge <- cv.glmnet(x = as.matrix(X_temp), y = y,
              alpha = 0, intercept = FALSE, family = "binomial")
          opt_lambda <- mod_ridge$lambda[which.min(mod_ridge$cvm)]
          coef_est <- as.vector(glmnet(x = as.matrix(X), y = y,
              alpha = 0, intercept = FALSE, family = "binomial",
              lambda = opt_lambda)$beta)
      }
      else {
          mod_lm <- glm(myform, data = mydata, family = "binomial")
          coef_est <- mod_lm$coefficients
      }
  }
  beta_refit <- rep(0, length(beta_pred))
  beta_refit[which(beta_pred != 0)] <- coef_est
  return(beta_refit)
}
```

test *WLogit output*

Description

The output of WLogit in the example given in the vignette.

Usage

```
data("test")
```

Format

The format is: List of 4 $ beta : num [1:50, 1:500] 0 0 0 0 0 ... $ lambda : num [1:50] 100.8 80 73 58.9 56.7 ... $ beta.min : num [1:500] 0.0194 0.0348 0.0259 0.0287 0.0385 ... $ log.likelihood: num [1:50] 57.7 57.7 57.7 57.7 57.7 ...

Examples

```
data(test)
str(test)
```

Thresholding *Thresholding on a vector*

Description

This function provides the thresholding (correction) given a vector. It calls the function top or top_thresh in the same package, and the output is the vector after correction with the optimal threshold parameter.

Usage

```
Thresholding(X, y, coef, TOP)
```

Arguments

X	Design matrix of the logistic model considered.
y	Binary response
coef	Candidate vector to be corrected
TOP	The grill of thresholding

Value

opt_top	The optimal threshold
auc	the log-likelihood for each grill of thresholding

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

top *Thresholding to zero of the smallest values*

Description

This function keeps only the K largest values of the vector sorted_vect and sets the others to zero.

Usage

```
top(vect, thresh)
```

Arguments

vect	vector to threshold
thresh	threshold

Value

This function returns the thresholded vector.

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

Examples

```
x=sample(1:10,10)
thresh=3
top(x,thresh)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (vect, thresh)
{
    sorted_vect <- sort(abs(vect), decreasing = TRUE)
    v = sorted_vect[thresh]
    ifelse(abs(vect) >= v, vect, 0)
  }
```

top_thresh *Thresholding to a given threshold of the smallest values*

Description

This function keeps only the K largest values of the vector `vect` and sets the others to the smallest value among the K largest.

Usage

```
top_thresh(vect, thresh)
```

Arguments

`vect`	vector to threshold
`thresh`	threshold

Value

This function returns the thresholded vector.

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

Examples

```
x=sample(1:10,10)
sorted_vect=sort(x,decreasing=TRUE)
thresh=3
top_thresh(x,thresh)

##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function (vect, thresh)
{
    sorted_vect <- sort(vect, decreasing = TRUE)
    v = sorted_vect[thresh]
    ifelse(vect >= v, vect, v)
  }
```

WhiteningLogit

Variable selection in high-dimensional logistic regression models using a whitening approach

Description

Variable selection in high-dimensional logistic regression models using a whitening approach

Usage

```
WhiteningLogit(X = X, y = y, nlambda = 50, maxit = 100, gamma = 0.9999,
top_grill=c(1:100))
```

Arguments

`X` Design matrix of the logistic model considered.

`y` Binary response of the logistic model considered.

`nlambda` Number of lambda

`maxit` Integer specifying the maximum number of steps for the generalized Lasso algorithm. It should not be smaller than nlambda.

`gamma` Parameter γ defined in the paper Zhu et al. (2022) given in the references. Its default value is 0.95.

`top_grill` A grill of provided for the thresholding

Value

Returns a list with the following components

`lambda` different values of the parameter λ considered.

`beta` matrix of the estimations of β for all the λ considered.

`beta.min` estimation of β which minimize the MSE.

`log.likelihood` Log-likelihood for all the λ considered.

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

References

W. Zhu, C. Levy-Leduc, N. Ternes. "Variable selection in high-dimensional logistic regression models using a whitening approach". (2022)

Examples

```
X0 <- matrix( rnorm(50*10,mean=0,sd=1), 50, 10)
y0 <- c(rep(1,25), rep(0,25))
mod <- WhiteningLogit(X=X0, y=y0)
plot(mod$beta.min)

##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(X=X, y=y,
                   nlambda=50,
                   maxit=100,
                   gamma=0.9999,
                   top_grill=c(1:100)){

  p=ncol(X)
  n=nrow(X)

  mod_ridge <- cv.glmnet(x=as.matrix(X), y=y, alpha=0.5, intercept=FALSE, family="binomial")
  pr_est <- predict(mod_ridge, as.matrix(X), s = "lambda.min", type="response")
  beta_ini <- predict(mod_ridge, as.matrix(X), s = "lambda.min", type="coefficients")[-1]
  diag_w <- pr_est*(1-pr_est)
  square_root_w <- diag(sqrt(as.vector(diag_w)), nrow=n)
  X_new <- square_root_w

  Cov_est <- cvCovEst(
    dat = X_new,
    estimators = c(
      linearShrinkLWEst, thresholdingEst, sampleCovEst
    ),
    estimator_params = list(
      thresholdingEst = list(gamma = seq(0.1, 0.3, 0.1))
    ),
    center = TRUE,
    scale = TRUE
  )

  Sigma_est <- Cov_est$estimate

  SVD_new <- fast.svd(Sigma_est)
  U_sigma_new <- SVD_new$u
  D_sigma_new <- SVD_new$d
  inv_transmat <- U_sigma_new
  inv_diag_new <- ifelse(D_sigma_new<0.000001, 0, 1/sqrt(D_sigma_new))
  trans_mat <- U_sigma_new


  if (p <= 50) {
    top_grill <- seq(1, p, 2)
  }else if (p <= 200) {
    top_grill <- c(1:50, seq(52, p, 2))
  }else if (p <= 300) {
    top_grill <- c(1:50, seq(52, 100, 2), seq(105, 200, 5),
                   seq(210, p, 10))
  }else {
    top_grill <- c(1:50, seq(52, 100, 2), seq(105, 200, 5),
                   seq(210, 300, 10))
  }

  X_tilde <- X

  beta_tilde_ini <-  inv_transmat
  Px <- CalculPx(X_tilde, beta=beta_tilde_ini)
  wt <- CalculWeight(Px)
  # wt <- ifelse(wt0==0, 0.0001, wt0)
  ystar <- WorkingResp(y=y, Px=Px, X=X_tilde, beta=beta_tilde_ini)
  X_tilde_weighted <- sweep(X, MARGIN=1, sqrt(wt), `*`)
  ystar_weighted <- sqrt(wt)*ystar

  gen.model0 <- genlasso(y=ystar_weighted, X=X_tilde_weighted,
                         D=trans_mat, maxsteps = 50)
  parameter_tmp <- beta_tilde_ini
  beta_final <- matrix(NA, length(gen.model0$lambda), p)
  skip_i <- TRUE
  eval_final <- c()
  defaultW <- getOption("warn")

  options(warn = -1)


  for(i in 1:length(gen.model0$lambda)){
    #inner loop
    epsilon=10
    j=0
```

```
if(skip_i){parameter_tmp <- beta_tilde_ini
} else {parameter_tmp <- parameter_current}
skip_i <-FALSE

while(epsilon > 0.001){
  j=j+1
  parameter_current <- parameter_tmp
  Px <- CalculPx(X_tilde, beta=parameter_current)
  wt0 <- CalculWeight(Px)
  wt <- ifelse(round(wt0,4)==0, 0.0001, wt)
  ystar <- WorkingResp(y=y, Px=Px, X=X_tilde, beta=parameter_current)
  X_tilde_weighted <- sweep(X, MARGIN=1, sqrt(wt), `*`)
  ystar_weighted <- sqrt(wt)*ystar

gen.model <- genlasso(y=ystar_weighted, X=X_tilde_weighted, D=trans_mat, maxsteps =  maxit)

  if(gen.model0$lambda[i] < min(gen.model$lambda)){
    parameter_tmp <- parameter_current
    break
  } else {
    parameter_tmp <- coef(gen.model, lambda=gen.model0$lambda[i],
                          type = "primal")$beta
    beta_current <- parameter_tmp
    if(sum(is.na(parameter_tmp))>0){
      skip_i <-TRUE
      parameter_tmp <- rep(0,p)
      break}
    epsilon <- max(abs(parameter_current-parameter_tmp))
    if(epsilon >=100){
      skip_i <-TRUE
      break}
    if (j==maxit){
      skip_i <-TRUE
      break}
  }
}

if(skip_i){
  beta_final[i, ] <- rep(NA, p)
  eval_final[i] <- NA
} else{

  correction <- Thresholding(X_tilde, y, coef=parameter_tmp, TOP=top_grill)
  opt_top_tilde <- correction$opt_top
  beta_tilde_opt <- top_thresh(vect=parameter_tmp, thresh = opt_top_tilde)
  beta_final0 <- trans_mat

  correction <- Thresholding(X, y, coef=beta_final0, TOP=top_grill)
  opt_top_final <- correction$opt_top
  beta_final[i, ] <- beta_opt_final <- top(vect=beta_final0, thresh = opt_top_final)

  beta_refit <- Refit_glm(X=X, beta_pred = beta_opt_final, y=y)
  pr_est <- CalculPx(X, beta_refit)
```

```
      ll <- pr_est^y*(1-pr_est)^(1-y)
      #ll <- ifelse(ll<0.000001, 1, ll)
      eval_final[i] <- -log(prod(ll))

    }
    beta.min <- beta_final[which.min(eval_final), ]
  }
  options(warn = defaultW)
  return(list(beta=beta_final, lambda=gen.model0$lambda, beta.min=beta.min,
              log.likelihood=eval_final))
}
```

`WorkingRespCalculate` *Calculate the working response*

Description

Calculate the working response in the iterative least square regression

Usage

```
WorkingResp(y, Px, X, beta, intercept = 0)
```

Arguments

`X`	Design matrix of the logistic model considered.
`y`	Binary response of the logistic model considered.
`Px`	The probability of the reponse to be 1
`beta`	Vector of coefficients
`intercept`	If there is an intercept

Value

This function returns the vector of working response.

Author(s)

Wencan Zhu, Celine Levy-Leduc, Nils Ternes

See Also

Please read https://hastie.su.domains/Papers/glmnet.pdf for more details

`X` *Example of a design matrix of a logistic model*

Description

It contains an example of a design matrix of a logistic model.

Usage

```
data("X")
```

Format

The format is: num [1:100, 1:500] -1.576 -0.476 -0.237 -0.398 0.284 ...

Examples

```
data(X)
```

`y` *Example of a binary response variable of a logistic model.*

Description

It contains an example of a binary response variable of a logistic model.

Usage

```
data("y")
```

Format

The format is: int [1:100] 0 1 0 1 1 0 0 0 1 1 ...

Examples

```
data(y)
```

Index