Package 'HTLR'

January 17, 2020

Version 0.4-2

Title Bayesian Logistic Regression with Heavy-Tailed Priors


License GPL-3

URL https://longhaisk.github.io/HTLR/

BugReports https://github.com/longhaiSK/HTLR/issues

Depends R (>= 3.1.0)

Suggests ggplot2, corrplot, testthat (>= 2.1.0), bayesplot, knitr, rmarkdown

Imports Rcpp (>= 0.12.0), BCBCSF, glmnet, magrittr

LinkingTo Rcpp (>= 0.12.0), RcppArmadillo

NeedsCompilation yes

SystemRequirements C++11

LazyData true

Encoding UTF-8

RoxygenNote 7.0.2

VignetteBuilder knitr

Author Longhai Li [aut, cre] (<https://orcid.org/0000-0002-3074-8584>), Steven Liu [aut]

Maintainer Longhai Li <longhai@math.usask.ca>

Repository CRAN

Date/Publication 2020-01-17 07:50:02 UTC
R topics documented:

- HTLR-package .................................................. 2
- as.matrix.htlr.fit .............................................. 3
- colon ............................................................. 3
- diabetes392 .................................................... 4
- evaluate_pred .................................................. 5
- gendata_FAM ................................................... 6
- gendata_MLR ................................................... 7
- htlr ............................................................... 8
- htlr_prior ....................................................... 10
- nzero_idx ....................................................... 11
- predict.htlr.fit .............................................. 12
- split_data ..................................................... 12
- std .............................................................. 13
- summary.htlr.fit ............................................ 14

Index 16

---

**HTLR-package**

Bayesian Logistic Regression with Heavy-Tailed Priors

**Description**

Efficient Bayesian multinomial logistic regression based on heavy-tailed priors. This package is suitable for classification and feature selection with high-dimensional features, such as gene expression profiles. Heavy-tailed priors can impose stronger shrinkage (compared to Gaussian and Laplace priors) on the coefficients associated with a large number of useless features, but still allow coefficients of a small number of useful features to stand out without punishment. It can also automatically make selection within a large number of correlated features. The posterior of coefficients and hyper-parameters is sampled with restricted Gibbs sampling for leveraging high-dimensionality and Hamiltonian Monte Carlo for handling high-correlations among coefficients.

**References**

as.matrix.htlr.fit

Create a Matrix of Markov Chain Samples

Description

The Markov chain samples (without warmup) included in a htlr.fit object will be coerced to a matrix.

Usage

## S3 method for class 'htlr.fit'
as.matrix(x, k = NULL, ...)

Arguments

x 
An object of S3 class htlr.fit.

k 
Coefficients associated with class k will be drawn. Must be a positive integer in 1,2,...,C-1 for C-class training labels (base class 0 can not be chosen). By default the last class is selected. For binary logistic model this argument can be ignored.

... 
Not used.

Value

A matrix with (p + 1) columns and i rows, where p is the number of features excluding intercept, and i is the number of iterations after burnin.

Examples

## No. of features used: 100; No. of iterations after burnin: 15
fit <- htlr(X = colon$X, y = colon$y, fsel = 1:100, iter = 20, warmup = 5)
dim(as.matrix(fit))

colon

Colon Tissues

Description

In this dataset, expression levels of 40 tumor and 22 normal colon tissues for 6500 human genes are measured using the Affymetrix technology. A selection of 2000 genes with highest minimal intensity across the samples has been made by Alon et al. (1999). The data is preprocessed by carrying out a base 10 logarithmic transformation and standardizing each tissue sample to zero mean and unit variance across the genes.
Usage
data("colon")

Format
A list contains data matrix \( X \) and response vector \( y \):

\[
\begin{align*}
X & \quad \text{A matrix with 66 rows (observations) and 2000 columns (features).} \\
y & \quad \text{A binary vector where 0 indicates normal colon tissues and 1 indicates tumor colon tissues.}
\end{align*}
\]

Source
ftp://stat.ethz.ch/Manuscripts/dettling/bagboost.pdf

References

---

diabetes392

Pima Indians Diabetes

Description
This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage. Different from the UCI original version, the dataset has been preprocessed such that rows with missing values are removed, and features are scaled.

Usage
data("diabetes392")

Format
A list contains data matrix \( X \) and response vector \( y \):

\[
\begin{align*}
X & \quad \text{A matrix with 392 rows (observations) and 8 columns (features).} \\
y & \quad \text{A binary vector where 1 indicates diabetes patients and 0 for otherwise.}
\end{align*}
\]

Source
References


See Also

https://avehtari.github.io/modelselection/diabetes.html

evaluate_pred

Evaluate Prediction Results

Description

This function compares the prediction results returned by a classifier with ground truth, and finally gives a summary of the evaluation.

Usage

evaluate_pred(y.pred, y.true, caseid = names(y.true), showplot = TRUE, ...)

Arguments

y.pred  A matrix of predicted probabilities, as returned by a classifier.
y.true  Ground truth labels vector.
caseid  The names of test cases which we take account of. By default all test cases are used for evaluation.
showplot  Logical; if TRUE, a summary plot will be generated.
...  Not used.

Value

A summary of evaluation result.
Description
This function generates inputs $X$ given by the response variable $y$ using a multivariate normal model.

Usage

gedata_FAM(n, muj, A, sd_g = 0, stdx = FALSE)

Arguments

- **n** Number of observations.
- **muj** C by p matrix, with row c representing $y=c$, and column j representing $x_j$. Used to specify $y$.
- **A** Factor loading matrix of size p by p, see details.
- **sd_g** Numeric value indicating noise level $\delta$, see details.
- **stdx** Logical; if TRUE, data $X$ is standardized to have mean = 0 and sd = 1.

Details
The means of each covariate $x_j$ depend on $y$ specified by the matrix muj; the covariate matrix $\Sigma$ of the multivariate normal is equal to $A A^t \delta^2 I$, where $A$ is the factor loading matrix and $\delta$ is the noise level.

Value
A list contains input matrix $X$, response variables $y$, covariate matrix $SGM$ and muj (standardized if stdx = TRUE).

See Also

gedata_MLR

Examples

```
## feature #1: marginally related feature
## feature #2: marginally unrelated feature, but feature #2 is correlated with feature #1
## feature #3-5: marginally related features and also internally correlated
## feature #6-10: noise features without relationship with the y

set.seed(12345)

n <- 100
p <- 10

means <- rbind(
c(0, 1, 0),
c(0, 0, 0),
c(0, 0, 1),
c(0, 0, 1),
c(0, 0, 1)
) * 2
means <- rbind(means, matrix(0, p - 5, 3))
A <- diag(1, p)
A[1:5, 1:3] <- rbind(
c(1, 0, 0),
c(2, 1, 0),
c(0, 0, 1),
c(0, 0, 1),
c(0, 0, 1)
)
dat <- gendata_FAM(n, means, A, sd_g = 0.5, stdx = TRUE)
ggplot2::qplot(dat$y, bins = 6)
corrplot::corrplot(cor(dat$X))

gendata_MLR

Generate Simulated Data with Multinomial Logistic Regression Model

Description
This function generates the response variables y given optional supplied X using a multinomial logistic regression model.

Usage
gendata_MLR(n, p, NC = 3, nu = 2, w = 1, X = NULL, betas = NULL)

Arguments
n Number of observations.
p Number of features.
NC Number of classes for response variables.
u, w If betas is not supplied (default), the regression coefficients are generated with t prior with df = nu, scale = sqrt(w); will be ignored if betas is supplied.
X The design matrix; will be generated from standard normal distribution if not supplied.
betas User supplied regression coefficients.

Value
A list contains input matrix X, response variables y, and regression coefficients deltas.
**htlr**

*Fit a HTLR Model*

**Description**

This function trains linear logistic regression models with HMC in restricted Gibbs sampling.

**Usage**

```r
htlr(
  X, y, fsel = 1:ncol(X), stdx = TRUE, prior = "t", df = 1, iter = 2000, warmup = floor(iter/2), thin = 1, init = "lasso", leap = 50, leap.warm = floor(leap/10), leap.stepsize = 0.3, cut = 0.05, verbose = FALSE, pre.legacy = FALSE, ...
)
```

**Arguments**

- `X` Input matrix, of dimension nobs by nvars; each row is an observation vector.
- `y` Vector of response variables. Must be coded as non-negative integers, e.g., 1,2,...,C for C classes, label 0 is also allowed.
- `fsel` Subsets of features selected before fitting, such as by univariate screening.

**See Also**

- `gendata_FAM`

**Examples**

```r
set.seed(12345)
dat <- gendata_MLR(n = 100, p = 10)
ggplot2::qplot(dat$y, bins = 6)
corrplot::corrplot(cor(dat$X))
```
stdx Logical; if TRUE, the original feature values are standardized to have mean = 0 and sd = 1.

prior The prior to be applied to the model. Either a list of hyperparameter settings returned by `htlr_prior` or a character string from "t" (student-t), "ghs" (horse-shoe), and "neg" (normal-exponential-gamma).

df The degree freedom of t/ghs/neg prior for coefficients. Will be ignored if the configuration list from `htlr_prior` is passed to prior.

iter A positive integer specifying the number of iterations (including warmup).

warmup A positive integer specifying the number of warmup (aka burnin). The number of warmup iterations should not be larger than iter and the default is iter / 2.

thin A positive integer specifying the period for saving samples.

init The initial state of Markov Chain; it accepts three forms:

• a previously fitted `fithtlr` object,
• a user supplied initial coefficient matrix of (p+1)*K, where p is the number of features, K is the number of classes in y minus 1,
• a character string matches the following:
  – "lasso" - (Default) Use Lasso initial state with lambda chosen by cross-validation. Users may specify their own candidate lambda values via optional argument `lasso.lambda`. Further customized Lasso initial states can be generated by `lasso_deltas`.
  – "bcbc" - Use initial state generated by package BCBCSF (Bias-corrected Bayesian classification). Further customized BCBCSF initial states can be generated by `bcbcsf_deltas`. WARNING: This type of initial states can be used for continuous features such as gene expression profiles, but it should not be used for categorical features such as SNP profiles.
  – "random" - Use random initial values sampled from N(0, 1).

leap The length of leapfrog trajectory in sampling phase.

leap.warm The length of leapfrog trajectory in burnin phase.

leap.stepsize The integrator step size used in the Hamiltonian simulation.

cut The coefficients smaller than this criteria will be fixed in each HMC updating step.

verbose Logical; setting it to TRUE for tracking MCMC sampling iterations.

pre.legacy Logical; if TRUE, the output produced in HTLR versions up to legacy-3.1-1 is reproduced. The speed will be typically slower than non-legacy mode on multicore machine. Default is FALSE.

... Other optional parameters:

• rda.alpha - A user supplied alpha value for `bcbcsf_deltas`. Default: 0.2.
• lasso.lambda - A user supplied lambda sequence for `lasso_deltas`. Default: {.01, .02, . . . , .05}. Will be ignored if `pre.legacy` is set to TRUE.

Value An object with S3 class `htlr.fit`. 
References

Examples
```r
set.seed(12345)
data("colon")

## fit HTLR models with selected features, note that the chain length setting is for demo only

## using t prior with 1 df and log-scale fixed to -10
fit.t <- htlr(X = colon$X, y = colon$y, fsel = 1:100,
               prior = htlr_prior("t", df = 1, logw = -10),
               init = "bcbc", iter = 20, thin = 1)

## using NEG prior with 1 df and log-scale fixed to -10
fit.neg <- htlr(X = colon$X, y = colon$y, fsel = 1:100,
                prior = htlr_prior("neg", df = 1, logw = -10),
                init = "bcbc", iter = 20, thin = 1)

## using horseshoe prior with 1 df and auto-selected log-scale
fit.ghs <- htlr(X = colon$X, y = colon$y, fsel = 1:100,
                 prior = "ghs", df = 1, init = "bcbc",
                 iter = 20, thin = 1)
```

htlr_prior

*Generate Prior Configuration*

**Description**
Configure prior hyper-parameters for HTLR model fitting

**Usage**
```
htlr_prior(
  ptype = c("t", "ghs", "neg"),
  df = 1,
  logw = -(1/df) * 10,
  sigmab0 = 2000
)
```

**Arguments**
- `ptype` The prior to be applied to the model. Either "t" (student-t, default), "ghs" (horseshoe), or "neg" (normal-exponential-gamma).
Details

The output is a configuration list which is to be passed to prior argument of htlr. For naive users, you only need to specify the prior type and degree freedom, then the other hyper-parameters will be chosen automatically. For advanced users, you can supply each prior hyper-parameters by yourself. For suggestion of picking hyper-parameters, see references.

Value

A configuration list containing ptype, alpha, logw, eta, and sigmab0.

References


nzero_idx

Get Indices of Non-Zero Coefficients

Description

Get the indices of non-zero coefficients from fitted HTLR model objects.

Usage

nzero_idx(fit, cut = 0.1)

Arguments

fit An object of S3 class htlr.fit.
cut Threshold on relative SDB to distinguish zero coefficients.

Value

Indices vector of non-zero coefficients in the model.

Examples

set.seed(12345)
data("colon")

fit <- htlr(X = colon$X, y = colon$y, fsel = 1:100, iter = 20)
nzero_idx(fit)
predict.htlr.fit  Make Prediction on New Data

Description

Similar to other predict methods, this function returns predictions from a fitted htlrfit object.

Usage

```r
## S3 method for class 'htlr.fit'
predict(object, newx, type = c("response", "class"), ...)
```

Arguments

- `object`: A fitted model object with S3 class htlrfit.
- `newx`: A Matrix of values at which predictions are to be made.
- `type`: Type of prediction required. Type "response" gives the fitted probabilities. Type "class" produces the class label corresponding to the maximum probability.
- `...`: Advanced options to specify the Markov chain iterations used for inference. See htlr_predict.

Value

The object returned depends on type.

split_data  Split Data into Train and Test Partitions

Description

This function splits the input data and response variables into training and testing parts.

Usage

```r
split_data(X, y, p.train = 0.7, n.train = round(nrow(X) * p.train))
```

Arguments

- `X`: Input matrix, of dimension nobs by nvars; each row is an observation vector.
- `y`: Vector of response variables.
- `p.train`: Percentage of training set.
- `n.train`: Number of cases for training; will override p.train if specified.
Value

List of training data \texttt{x.tr}, \texttt{y.tr} and testing data \texttt{x.te}, \texttt{y.te}.

Examples

dat <- gendata_MLR(n = 100, p = 10)
dat <- split_data(dat$X, dat$y, p.train = 0.7)
dim(dat$x.tr)
dim(dat$x.te)

---

**std**

*Standardizes a Design Matrix*

### Description

This function accepts a design matrix and returns a standardized version of that matrix, the statistics of each column such as median and sd are also provided.

### Usage

```r
std(X, tol = 1e-06)
```

### Arguments

- **X**: Design matrix, of dimension \texttt{nobs} by \texttt{nvars}; each row is an observation vector; can also be an object that can be coerced to a matrix, e.g. a data.frame.
- **tol**: The tolerance value; a column of \texttt{X} is considered as singular if the \texttt{sd} of its entries (observations) is less than \texttt{tol}. Singular columns will be dropped by the end.

### Details

For each column of \texttt{X}, the standardization is done by first subtracting its median, then dividing by its sample standard deviation, while the original version in \texttt{ncvreg} uses mean and population standard deviation. Its speed is slower than \texttt{ncvreg} because of the complexity of median finding, but still substantially faster than \texttt{scale()} provided by R base.

### Value

The standardized design matrix with the following attributes:

- **nonsingular**: Indices of non-singular columns.
- **center**: Median of each non-singular column which is used for standardization.
- **scale**: Standard deviation of each non-singular column which is used for standardization.
summary.htlr.fit

Author(s)

Patrick Breheny (original)
Steven Liu (modification)

See Also


Examples

```r
set.seed(123)
mat <- matrix(rnorm(n = 80 * 90, mean = 100, sd = 50), 80, 90)
mat %>% as.numeric() %>% ggplot2::qplot(bins = 30, xlab = quot;Varquot;)
mat %>% std() %>% as.numeric() %>% ggplot2::qplot(bins = 30, xlab = quot;'
```

summary.htlr.fit Posterior Summaries

Description

This function gives a summary of posterior of parameters.

Usage

```r
## S3 method for class 'htlr.fit'
summary(object, features = 1L:object$p, method = median, usedmc = NULL, ...)
```

Arguments

- **object**
  An object of S3 class htlr.fit.

- **features**
  A vector of indices (int) or names (char) that specify the parameters we will look at. By default all parameters are selected.

- **method**
  A function that is used to aggregate the MCMC samples. The default is median, other built-in/customized statistical functions such as mean, sd, and mad can also be used.

- **usedmc**
  Indices of Markov chain iterations used for inference. By default all iterations are used.

- ...
  Not used.

Value

A point summary of MCMC samples.
Examples

set.seed(12345)
data("colon")

fit <- htlr(X = colon$X, y = colon$y, fsel = 1:100, iter = 20)
summary(fit, features = 1:16)
# Index

*Topic **datasets**
  
thlr, 8
HTLR-package, 2
htlr_predict, 12
htlr_prior, 9, 10

topic, datasets
  
colon, 3
  
diabetes392, 4
  
as.matrix.htlr.fit, 3
  
bcbscf_deltas, 9
  
 colon, 3
  
diabetes392, 4
  
evaluate_pred, 5
  
gendata_FAM, 6, 8
  
gendata_MLR, 6, 7
  
htlr, 8
HTLR-package, 2
htlr_predict, 12
htlr_prior, 9, 10

lasso_deltas, 9

nzero_idx, 11

predict.htlr.fit, 12

split_data, 12

std, 13

summary.htlr.fit, 14