Package 'HTLR'

December 27, 2017

Version 3.1-1

Title Logistic Regression Based on Heavy-Tailed Priors

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

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

Depends R (>= 2.10.1), glmnet

Description This package performs classification and feature selection by fitting Baeysian polychotomous (multiclass) logistic regression models based on heavy-tailed priors with small degree freedom. The software is suitable for classification with high-dimensional features, such as gene expression profiles. Heavy-tailed priors can impose stronger shrinkage (compared to Guassian and Laplace priors) to 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. Heavy-tailed priors can also automatically make selection within a large number of correlated features. The posterior of coefficients and hyperparameters is sampled with resitricted Gibbs sampling for leveraging high-dimensionality and Hamiltonian Monte Carlo for handling high-correlations among coefficients. The core computation in this software is carried out with fast C code.

License GPL (>=2)

URL http://www.r-project.org, http://math.usask.ca/~longhai

R topics documented:

d0:demo 1
d1:fitting 3

`d0:demo` *Examples of Using HTLR*

Description

This help file gives a step-by-step example of using HTLR package how to fit HTLR model, select feature subsets, look at selected feature subsets, plot feature importance score, and make prediction for test cases with fitting results.

Examples

```
# load library
library (HTLR, lib.loc = "~/Rdev/HTLR_3.1-1")

#############################################################################
######################### create/load datasets ##############################
#############################################################################

## generate a dataset with grouping structure
## to analyze read data, replace the following objects with your data matrix/vector
source ("gen_jscs_data.R")
## split into training and testing datasets
ntr <- 100
X_tr <- data$X[1:ntr, ]
y_tr <- data$y[1:ntr]
X_ts <- data$X[-(1:ntr), ]
y_ts <- data$y[-(1:ntr)]


#############################################################################
######################### Model fitting and feature selection ###############
#############################################################################

######################### Fit LASSO  ########################################
## (this step can be used to pre-select features for dataset with dim > 2000)

lfit <- lasso_fitpred (X_tr, y_tr, X_ts) ## fit lasso and make predictions on test cases

## looking at coefficients and feature selection
sdb.lfit <- comp_sdb (lfit$deltas, normalize = F)
## draw lasso coefficients for visualization
plot_fscore (sdb.lfit, log = "x", main = "LASSO")
sel.lfit <- which(sdb.lfit > 0.1 * max (sdb.lfit)); sel.lfit
length (sel.lfit)

################### Fit HTLR with t prior ###################################
tfit <- htlr_fit (
    y_tr = y_tr, X_tr = X_tr, X_ts = X_ts, stdzx = T, fsel = 1:ncol(X_tr),  ##  data
    pty = "t", alpha = 1, s = -10,  ## alpha = df and s= log (w)
    iters_h = 1000, iters_rmc = 1000, thin = 1, ## mcmc iteration settings,
    leap_L_h = 5, leap_L = 50, leap_step = 0.5, hmc_sgmcut = 0.05, ## hmc settings
    initial_state = "bcbcsfrda", silence = F)  ## initial state

## looking at coefficients and feature selection
sdb.tfit <- htlr_sdb(tfit)
plot_fscore (sdb.tfit, log = "x", main = "Cauchy")

sel.tfit <- which(sdb.tfit > 0.1*max(sdb.tfit)); sel.tfit
length (sel.tfit)

################### Fit HTLR with horseshoe prior ############################

hfit <- htlr_fit (
    y_tr = y_tr, X_tr = X_tr, X_ts = X_ts, stdzx = T, fsel = 1:ncol(X_tr),  ##  data
    pty = "ghs", alpha = 1, s = -10,  ## alpha = df and s= log (w)
```

```
    iters_h = 1000, iters_rmc = 1000, thin = 1, ## mcmc iteration settings,
    leap_L_h = 5, leap_L = 50, leap_step = 0.5, hmc_sgmcut = 0.05, ## hmc settings
    initial_state = "bcbcsfrda", silence = F)  ## initial state

## looking at coefficients and feature selection
sdb.hfit <- htlr_sdb(hfit)
plot_fscore (sdb.hfit, log = "x", main = "Horseshoe")

sel.hfit <- which(sdb.hfit > 0.1*max(sdb.hfit)); sel.hfit
length (sel.hfit)

################### Fit HTLR with NEG prior ############################

nfit <- htlr_fit (
    y_tr = y_tr, X_tr = X_tr, X_ts = X_ts, stdzx = T, fsel = 1:ncol(X_tr),  ##  data
    pty = "neg", alpha = 1, s = -10,  ## alpha = df and s= log (w)
    iters_h = 1000, iters_rmc = 1000, thin = 1, ## mcmc iteration settings,
    leap_L_h = 5, leap_L = 50, leap_step = 0.5, hmc_sgmcut = 0.05, ## hmc settings
    initial_state = "bcbcsfrda", silence = F)  ## initial state

## looking at coefficients and feature selection
sdb.nfit <- htlr_sdb(nfit)
plot_fscore (sdb.nfit, log = "x", main = "NEG")

sel.nfit <- which(sdb.nfit > 0.1*max(sdb.nfit)); sel.nfit
length (sel.nfit)

#############################################################################
######################### Out-of-sample Prediction Comparison ###############
#############################################################################

#Note: predictions have been produced in fitting functions

## evaluate predictions on test cases (out-of-sample testing) by lasso
lpred <- lfit$probs_pred ## prediction results on test cases
## evaluate lasso predictions with ER and AMLP
evaluate_pred (lpred, y_ts, method = "LASSO") -> lpred.eval

## evaluate tprior predictions with ER and AMLP
bpred <- tfit$probs_pred
evaluate_pred (bpred, y_ts, method = "Caucy") -> bpred.eval
## evaluate horseshoe predictions with ER and AMLP
hpred <- hfit$probs_pred
evaluate_pred (hpred, y_ts, method = "Horseshoe") -> hpred.eval
## evaluateneg predictions with ER and AMLP
npred <- nfit$probs_pred
evaluate_pred (npred, y_ts,method = "NEG") -> npred.eval
```

`d1:fitting` *Fitting HTLR Models and Making Predictions*

Description

`htlr_fit` trains linear logistic regression models with HMC in restricted Gibbs sampling. This function also makes predictions for test cases if `X_ts` are provided.

htlr_predict uses MCMC samples returned by htlr_fit to predict the class labels of test cases. Prediction results are a matrix of predictive probabilities and a vector of predicted class labels

Usage

```
htlr_fit (
    y_tr, X_tr, X_ts = NULL, fsel = 1:ncol(X_tr), stdzx = TRUE,
    sigmab0 = 2000, ptype = "t", alpha = 1, s = -15, eta = 0,
    iters_h = 2000, iters_rmc = 2000, thin = 10,
    leap_L = 50, leap_L_h = 5, leap_step = 0.5,  hmc_sgmcut = 0.05,
    looklf = 0,  initial_state = NULL, silence = TRUE,
    predburn = NULL, predthin = 1)

htlr_predict (X_ts, fithtlr = NULL, deltas = NULL,
              burn = NULL, thin = NULL, usedmc = NULL)
```

Arguments

X_tr, X_ts	matrices containing data; rows should be for the cases, and columns for different features; X_tr are training data, X_ts are test data or future data for which prediction are needed.
fsel	subsets of features selected before fitting, such as by univariate screening.
stdzx	if it is set to TRUE, the original features values are standardized to have mean 0 and sd 1 for each gene; by default, it is TRUE.
y_tr,y	a vector of class labels in training or test data set. Must be coded as positive integers 1,2,...,C for C classes.
iters_h, iters_rmc, thin	iters_h and iters_rmc of super Markov chain transitions, each with thin Markov chain iterations, are run for burning (aka head/initial) and sampling phases; only the last state of each super transition in the sampling phase is saved.
alpha, s, sigmab0, ptype, eta	Prior settings for coefficients. alpha is the degree freedom, and s is log (w), equal to twice of log scale of priors for coefficients.
silence	logical. Setting it to TRUE for tracking MCMC sampling iterations.
fithtlr	a list containing fitting results by htlr_fit
predburn,predthin	predburn of Markov chain (super)iterations will be discarded for prediction, and only every predthinth are used; by default, 20% of (super)iterations are burned, and thin=1.
burn, thin	the meanings are the same as predburn and predthin

Value

htlr_fit returns a list of fitting results. htlr_predict returns a matrix of predictive probabilities, with rows for cases, cols for classes.

Index

∗Topic **classif**
    d1:fitting, 3

d0:demo, 1
d1:fitting, 3

htlr:demonstration *(d0:demo)*, 1
htlr_fit *(d1:fitting)*, 3
htlr_predict *(d1:fitting)*, 3