Package 'hal9001'

June 27, 2020

Title The Scalable Highly Adaptive Lasso

Version 0.2.6

Description A scalable implementation of the highly adaptive lasso algorithm, including routines for constructing sparse matrices of basis functions of the observed data, as well as a custom implementation of Lasso regression tailored to enhance efficiency when the matrix of predictors is composed exclusively of indicator functions. For ease of use and increased flexibility, the Lasso fitting routines invoke code from the 'glmnet' package by default. The highly adaptive lasso was first formulated and described by MJ van der Laan (2017) <doi:10.1515/ijb-2015-0097>, with practical demonstrations of its performance given by Benkeser and van der Laan (2016) <doi:10.1109/DSAA.2016.93>.

Depends R (>= 3.1.0), Rcpp

License GPL-3

URL https://github.com/tlverse/hal9001

BugReports https://github.com/tlverse/hal9001/issues

Encoding UTF-8

LazyData true

Imports Matrix, stats, utils, methods, assertthat, origami (>= 1.0.3), glmnet

Suggests testthat, knitr, rmarkdown, microbenchmark, future, ggplot2, dplyr, tidyr, stringr, survival, data.table, SuperLearner

LinkingTo Rcpp, RcppEigen

VignetteBuilder knitr

RoxygenNote 7.1.0

NeedsCompilation yes

Author Jeremy Coyle [aut, cre] (<https://orcid.org/0000-0002-9874-6649>), Nima Hejazi [aut] (<https://orcid.org/0000-0002-7127-2789>), David Benkeser [ctb] (<https://orcid.org/0000-0002-1019-8343>), Oleg Sofrygin [ctb], Weixin Cai [ctb] (<https://orcid.org/0000-0003-2680-3066>), Mark van der Laan [aut, cph, ths] (<https://orcid.org/0000-0003-1432-5511>)

Maintainer Jeremy Coyle <jeremyrcoyle@gmail.com>

Repository CRAN

Date/Publication 2020-06-27 04:50:07 UTC

R topics documented:

`apply_copy_map` *Apply copy map*

Description

OR duplicate training set columns together

Usage

```
apply_copy_map(X, copy_map)
```

Arguments

`X`	Sparse matrix containing columns of indicator functions.
`copy_map`	the copy map

Value

A dgCMatrix sparse matrix corresponding to the design matrix for a zero-th order highly adaptive lasso, but with all duplicated columns (basis functions) removed.

Examples

```
gendata <- function(n) {
  W1 <- runif(n, -3, 3)
  W2 <- rnorm(n)
  W3 <- runif(n)
  W4 <- rnorm(n)
  g0 <- plogis(0.5 * (-0.8 * W1 + 0.39 * W2 + 0.08 * W3 - 0.12 * W4))
  A <- rbinom(n, 1, g0)
  Q0 <- plogis(0.15 * (2 * A + 2 * A * W1 + 6 * A * W3 * W4 - 3))
  Y <- rbinom(n, 1, Q0)
  data.frame(A, W1, W2, W3, W4, Y)
}
set.seed(1234)
data <- gendata(100)
covars <- setdiff(names(data), "Y")
X <- as.matrix(data[, covars, drop = FALSE])
basis_list <- enumerate_basis(X)
x_basis <- make_design_matrix(X, basis_list)
copy_map <- make_copy_map(x_basis)
x_basis_uniq <- apply_copy_map(x_basis, copy_map)
```

---

as_dgCMatrix *Fast Coercion to Sparse Matrix*

Description

Fast and efficient coercion of standard matrix objects to sparse matrices. Borrowed from http://gallery.rcpp.org/articles/sparse-matrix-coercion/. INTERNAL USE ONLY.

Usage

```
as_dgCMatrix(XX_)
```

Arguments

XX_	An object of class Matrix that has a sparse structure suitable for coercion to a sparse matrix format of dgCMatrix.

Value

An object of class dgCMatrix, coerced from input XX_.

basis_list_cols *List Basis Functions*

Description

Build a list of basis functions from a set of columns

Usage

```
basis_list_cols(cols, x)
```

Arguments

cols	Index or indices (as numeric) of covariates (columns) of interest in the data matrix x for which basis functions ought to be generated. Note that basis functions for interactions of these columns are computed automatically.
x	A matrix containing observations in the rows and covariates in the columns. Basis functions are computed for these covariates.

Value

A list containing the basis functions generated from a set of input columns.

basis_of_degree *Compute Degree of Basis Functions*

Description

Find the full list of basis functions up to a particular degree

Usage

```
basis_of_degree(x, degree)
```

Arguments

x	An input matrix containing observations and covariates following standard conventions in problems of statistical learning.
degree	The highest order of interaction terms for which the basis functions ought to be generated. The default (NULL) corresponds to generating basis functions for the full dimensionality of the input matrix.

Value

A list containing basis functions and cutoffs generated from a set of input columns up to a particular pre-specified degree.

cv_lasso *Cross-validated Lasso on Indicator Bases*

Description

Fits Lasso regression using a customized procedure, with cross-validation based on **origami**

Usage

```
cv_lasso(x_basis, y, n_lambda = 100, n_folds = 10, center = FALSE)
```

Arguments

x_basis	A dgCMatrix object corresponding to a sparse matrix of the basis functions generated for the HAL algorithm.
y	A numeric vector of the observed outcome variable values.
n_lambda	A numeric scalar indicating the number of values of the L1 regularization parameter (lambda) to be obtained from fitting the Lasso to the full data. Cross-validation is used to select an optimal lambda (that minimizes the risk) from among these.
n_folds	A numeric scalar for the number of folds to be used in the cross-validation procedure to select an optimal value of lambda.
center	binary. If TRUE, covariates are centered. This is much slower, but matches the glmnet implementation. Default FALSE.

cv_lasso_early_stopping *Cross-validated LASSO on Indicator Bases*

Description

Fits the LASSO regression using a customized procedure with cross-validation based on **origami**

Usage

```
cv_lasso_early_stopping(x_basis, y, n_lambda = 100, n_folds = 10)
```

Arguments

x_basis	A dgCMatrix object corresponding to a sparse matrix of the basis functions generated for the HAL algorithm.
y	A numeric vector of the observed outcome variable values.

n_lambda	A numeric scalar indicating the number of values of the L1 regularization parameter (lambda) to be obtained from fitting the LASSO to the full data. Cross-validation is used to select an optimal lambda (that minimizes the risk) from among these.
n_folds	A numeric scalar for the number of folds to be used in the cross-validation procedure to select an optimal value of lambda.

---

enumerate_basis *Enumerate Basis Functions*

Description

Generate basis functions for all covariates and interaction terms thereof up to a specified order/degree

Usage

```
enumerate_basis(x, max_degree = NULL)
```

Arguments

x	An input matrix containing observations and covariates following standard conventions in problems of statistical learning.
max_degree	The highest order of interaction terms for which the basis functions ought to be generated. The default (NULL) corresponds to generating basis functions for the full dimensionality of the input matrix.

Value

A list of basis functions generated for all covariates and interaction thereof up to a pre-specified degree.

Examples

```
gendata <- function(n) {
  W1 <- runif(n, -3, 3)
  W2 <- rnorm(n)
  W3 <- runif(n)
  W4 <- rnorm(n)
  g0 <- plogis(0.5 * (-0.8 * W1 + 0.39 * W2 + 0.08 * W3 - 0.12 * W4))
  A <- rbinom(n, 1, g0)
  Q0 <- plogis(0.15 * (2 * A + 2 * A * W1 + 6 * A * W3 * W4 - 3))
  Y <- rbinom(n, 1, Q0)
  data.frame(A, W1, W2, W3, W4, Y)
}
set.seed(1234)
data <- gendata(100)
covars <- setdiff(names(data), "Y")
```

```
X <- as.matrix(data[, covars, drop = FALSE])
basis_list <- enumerate_basis(X)
```

`evaluate_basis` *Generate Basis Functions*

Description

Populates a column (indexed by basis_col) of x_basis with basis indicators.

Usage

```
evaluate_basis(basis, X, x_basis, basis_col)
```

Arguments

`basis`	The basis function.
`X`	The design matrix, containing the original data.
`x_basis`	The HAL design matrix, containing indicator functions.
`basis_col`	Numeric indicating which column to populate.

`fit_hal` *HAL: The Highly Adaptive Lasso*

Description

Estimation procedure for HAL, the Highly Adaptive Lasso

Usage

```
fit_hal(
  X,
  Y,
  X_unpenalized = NULL,
  max_degree = 3,
  fit_type = c("glmnet", "lassi"),
  n_folds = 10,
  foldid = NULL,
  use_min = TRUE,
  reduce_basis = NULL,
  family = c("gaussian", "binomial", "cox"),
  return_lasso = TRUE,
  return_x_basis = FALSE,
```

```
  basis_list = NULL,
  lambda = NULL,
  id = NULL,
  offset = NULL,
  cv_select = TRUE,
  ...,
  yolo = TRUE
)
```

Arguments

X	An input matrix containing observations and covariates.
Y	A numeric vector of obervations of the outcome variable.
X_unpenalized	An input matrix with the same format as X, that directly get appended into the design matrix (no basis expansion). No L-1 penalization is performed on these covariates.
max_degree	The highest order of interaction terms for which the basis functions ought to be generated. The default (NULL) corresponds to generating basis functions for the full dimensionality of the input matrix.
fit_type	The specific routine to be called when fitting the Lasso regression in a cross-validated manner. Choosing the glmnet option will result in a call to cv.glmnet while lassi will produce a (faster) call to a custom Lasso routine.
n_folds	Integer for the number of folds to be used when splitting the data for V-fold cross-validation. This defaults to 10.
foldid	An optional vector of values between 1 and n_folds identifying what fold each observation is in. If supplied, n_folds can be missing. When supplied, this is passed to cv.glmnet.
use_min	Determines which lambda is selected from cv.glmnet. TRUE corresponds to "lambda.min" and FALSE corresponds to "lambda.1se".
reduce_basis	A numeric value bounded in the open interval (0,1) indicating the minimum proportion of 1's in a basis function column needed for the basis function to be included in the procedure to fit the Lasso. Any basis functions with a lower proportion of 1's than the cutoff will be removed. This argument defaults to NULL, in which case all basis functions are used in the lasso-fitting stage of the HAL algorithm.
family	A character corresponding to the error family for a generalized linear model. Options are limited to "gaussian" for fitting a standard linear model, "binomial" for penalized logistic regression, "cox" for a penalized proportional hazards model. Note that in the case of "binomial" and "cox" the argument fit_type is limited to "glmnet"; thus, documentation of the glmnet package should be consulted for any errors resulting from the Lasso fitting step in these cases.
return_lasso	A logical indicating whether or not to return the glmnet fit of the lasso model.
return_x_basis	A logical indicating whether or not to return the matrix of (possibly reduced) basis functions used in the HAL lasso fit.

basis_list	The full set of basis functions generated from the input data X (via a call to enumerate_basis). The dimensionality of this structure is dim = (n * 2^(d - 1)), where n is the number of observations and d is the number of columns in X.
lambda	User-specified array of values of the lambda tuning parameter of the Lasso L1 regression. If NULL, cv.glmnet will be used to automatically select a CV-optimal value of this regularization parameter. If specified, the Lasso L1 regression model will be fit via glmnet, returning regularized coefficient values for each value in the input array.
id	a vector of ID values, used to generate cross-validation folds for cross-validated selection of the regularization parameter lambda.
offset	a vector of offset values, used in fitting.
cv_select	A logical specifying whether the array of values specified should be passed to cv.glmnet in order to pick the optimal value (based on cross-validation) (when set to TRUE) or to simply fit along the sequence of values (or single value) using glmnet (when set to FALSE).
...	Other arguments passed to cv.glmnet. Please consult its documentation for a full list of options.
yolo	A logical indicating whether to print one of a curated selection of quotes from the HAL9000 computer, from the critically acclaimed epic science-fiction film "2001: A Space Odyssey" (1968).

Details

The procedure uses a custom C++ implementation to generate a design matrix consisting of basis functions corresponding to covariates and interactions of covariates and to remove duplicate columns of indicators. The Lasso regression is fit to this (usually) very wide matrix using either a custom implementation (based on **origami**) or by a call to cv.glmnet.

Value

Object of class hal9001, containing a list of basis functions, a copy map, coefficients estimated for basis functions, and timing results (for assessing computational efficiency).

Examples

```
n <- 100
p <- 3
x <- xmat <- matrix(rnorm(n * p), n, p)
y_prob <- plogis(3 * sin(x[, 1]) + sin(x[, 2]))
y <- rbinom(n = n, size = 1, prob = y_prob)
ml_hal_fit <- fit_hal(X = x, Y = y, family = "binomial", yolo = FALSE)
preds <- predict(ml_hal_fit, new_data = x)
```

`hal9000` *HAL 9000 Quotes*

Description

Prints a quote from the HAL 9000 robot from 2001: A Space Odyssey

Usage

```
hal9000()
```

`hal9001` *hal9001*

Description

Package for fitting the Highly Adaptive LASSO (HAL) estimator

`hal_quotes` *HAL9000 Quotes from "2001: A Space Odyssey"*

Description

Curated selection of quotes from the HAL9000 computer, from the critically acclaimed epic science-fiction film "2001: A Space Odyssey" (1968).

Usage

```
hal_quotes
```

Format

A vector of quotes.

index_first_copy *Find Copies of Columns*

Description

Index vector that, for each column in X, indicates the index of the first copy of that column

Usage

```
index_first_copy(X)
```

Arguments

X	Sparse matrix containing columns of indicator functions.

lassi_fit_module *Rcpp module: lassi_fit_module*

Description

Rcpp module: lassi_fit_module

lassi_origami *Single Lasso estimation for cross-validation with Origami*

Description

Fits Lasso regression over a single fold of a cross-validated data set. This is meant to be called using cross_validate, which is done through cv_lasso. Note that this procedure is NOT meant to be invoked by itself. INTERNAL USE ONLY.

Usage

```
lassi_origami(fold, data, lambdas, center = FALSE)
```

Arguments

fold	A fold object produced by a call to make_folds from the **origami**.
data	A dgCMatrix object containing the outcome values (Y) in its first column and vectors corresponding to the basis functions of HAL in all other columns. Consult the description of HAL regression for details.
lambdas	A numeric vector corresponding to a sequence of lambda values obtained by fitting the Lasso on the full data.
center	binary. If TRUE, covariates are centered. This is much slower, but matches the glmnet implementation. Default FALSE.

make_basis_list *Sort Basis Functions*

Description

Build a sorted list of unique basis functions based on columns, where each basis function is a list

Usage

```
make_basis_list(X_sub, cols)
```

Arguments

X_sub	A subset of the columns of X, the original design matrix.
cols	An index of the columns that were reduced to by sub-setting.

Details

Note that sorting of columns is performed such that the basis order equals cols.length() and each basis function is a list(cols, cutoffs).

make_copy_map *Build Copy Maps*

Description

Build Copy Maps

Usage

```
make_copy_map(x_basis)
```

Arguments

x_basis	A design matrix consisting of basis (indicator) functions for covariates (X) and terms for interactions thereof.

Value

A `list` of `numeric` vectors indicating indices of basis functions that are identical in the training set.

Examples

```
gendata <- function(n) {
  W1 <- runif(n, -3, 3)
  W2 <- rnorm(n)
  W3 <- runif(n)
  W4 <- rnorm(n)
  g0 <- plogis(0.5 * (-0.8 * W1 + 0.39 * W2 + 0.08 * W3 - 0.12 * W4))
  A <- rbinom(n, 1, g0)
  Q0 <- plogis(0.15 * (2 * A + 2 * A * W1 + 6 * A * W3 * W4 - 3))
  Y <- rbinom(n, 1, Q0)
  data.frame(A, W1, W2, W3, W4, Y)
}
set.seed(1234)
data <- gendata(100)
covars <- setdiff(names(data), "Y")
X <- as.matrix(data[, covars, drop = FALSE])
basis_list <- enumerate_basis(X)
x_basis <- make_design_matrix(X, basis_list)
copy_map <- make_copy_map(x_basis)
```

---

make_design_matrix *Build HAL Design Matrix*

Description

Make a HAL design matrix based on original design matrix X and a list of basis functions in argument blist

Usage

```
make_design_matrix(X, blist)
```

Arguments

X	Matrix of covariates containing observed data in the columns.
blist	List of basis functions with which to build HAL design matrix.

Value

A dgCMatrix sparse matrix of indicator basis functions corresponding to the design matrix in a zero-order highly adaptive lasso.

Examples

```
gendata <- function(n) {
  W1 <- runif(n, -3, 3)
  W2 <- rnorm(n)
  W3 <- runif(n)
  W4 <- rnorm(n)
  g0 <- plogis(0.5 * (-0.8 * W1 + 0.39 * W2 + 0.08 * W3 - 0.12 * W4))
  A <- rbinom(n, 1, g0)
  Q0 <- plogis(0.15 * (2 * A + 2 * A * W1 + 6 * A * W3 * W4 - 3))
  Y <- rbinom(n, 1, Q0)
  data.frame(A, W1, W2, W3, W4, Y)
}
set.seed(1234)
data <- gendata(100)
covars <- setdiff(names(data), "Y")
X <- as.matrix(data[, covars, drop = FALSE])
basis_list <- enumerate_basis(X)
x_basis <- make_design_matrix(X, basis_list)
```

---

meets_basis *Compute Values of Basis Functions*

Description

Computes and returns the indicator value for the basis described by cols and cutoffs for a given row of X (X[row_num,])

Usage

```
meets_basis(X, row_num, cols, cutoffs)
```

Arguments

X	The design matrix, containing the original data.
row_num	Numeri for a row index over which to evaluate.
cols	Numeric for the column indices of the basis function.
cutoffs	Numeric providing thresholds.

`predict.hal9001` *Prediction from HAL fits*

Description

Prediction from HAL fits

Usage

```
## S3 method for class 'hal9001'
predict(object, offset = NULL, ..., new_data, new_X_unpenalized = NULL)
```

Arguments

`object`	An object of class `hal9001`, containing the results of fitting the Highly Adaptive Lasso, as produced by `fit_hal`.
`offset`	A vector of offsets. Must be provided if provided at training
`...`	Additional arguments passed to `predict` as necessary.
`new_data`	A `matrix` or `data.frame` containing new data (observations NOT used in fitting the `hal9001` object passed in via the `object` argument above) for which the `hal9001` object will compute predicted values.
`new_X_unpenalized`	If the user supplied `X_unpenalized` during training, the user should also supply this matrix with the same number of observations as `new_data`. Optional.

Details

Method for computing and extracting predictions from fits of the Highly Adaptive Lasso estimator, returned as a single S3 objects of class `hal9001`.

Value

A `numeric` vector of predictions from a `hal9001` object.

Note

This prediction method does not function similarly to the equivalent method from **glmnet**. In particular, this procedure will NOT return a subset of lambdas originally specified in callingo `fit_hal` nor result in re-fitting. Instead, it will return predictions for all of the lambdas specified in the call to `fit_hal` that constructs `object`, when `cv_select = FALSE`. When `cv_select = TRUE`, predictions will only be returned for the value of lambda selected by cross-validation.

predict.SL.hal9001 *predict.SL.hal9001*

Description

Predict method for objects of class `SL.hal9001`

Usage

```
## S3 method for class 'SL.hal9001'
predict(object, newdata, ...)
```

Arguments

`object`	A fitted object of class `hal9001`.
`newdata`	A matrix of new observations on which to obtain predictions.
`...`	Placeholder (ignored).

Value

A `numeric` vector of predictions from a `SL.hal9001` object based on the provide `newdata`.

SL.hal9001 *Wrapper for Classic SuperLearner*

Description

Wrapper for **SuperLearner** for objects of class `hal9001`

Usage

```
SL.hal9001(
  Y,
  X,
  newX = NULL,
  max_degree = 3,
  fit_type = c("glmnet", "lassi"),
  n_folds = 10,
  use_min = TRUE,
  family = stats::gaussian(),
  obsWeights = rep(1, length(Y)),
  ...
)
```

Arguments

Y	A numeric of outcomes.
X	A matrix of predictors/covariates.
newX	A matrix of new observations on which to obtain predictions. The default of NULL computes predictions on training inputs X.
max_degree	The highest order of interaction terms for which the basis functions ought to be generated. NULL corresponds to generating basis functions for the full dimensionality of the input matrix.
fit_type	The specific routine to be called when fitting the Lasso regression via cross-validation. Choosing cv.glmnet option results in option results in a call to cv.glmnet while lassi produces a (faster) call to a custom routine based on a custom routine for fitting the Lasso.
n_folds	Integer for the number of folds to be used when splitting the data for cross-validation. This defaults to 10 as this is the convention for V-fold cross-validation.
use_min	Determines which lambda is selected from cv.glmnet. TRUE corresponds to "lambda.min" and FALSE corresponds to "lambda.1se".
family	Not used by the function directly, but meant to ensure compatibility with SuperLearner.
obsWeights	Not used by the function directly, but meant to ensure compatibility with SuperLearner. These are passed to cv.glmnet through the ... argument of fit_hal.
...	Placeholder (ignored).

Value

An object of class SL.hal9001 with a fitted hal9001 object and corresponding predictions based on the input data.

---

squash_hal_fit *Squash HAL objects*

Description

Reduce footprint by dropping basis functions with coefficients of zero

Usage

```
squash_hal_fit(object)
```

Arguments

object	An object of class hal9001, containing the results of fitting the Highly Adaptive LASSO, as produced by a call to fit_hal.

Value

Object of class `hal9001`, similar to the input object but reduced such that coefficients belonging to bases with coefficients equal to zero removed.

Examples

```
# generate simple test data
n <- 100
p <- 3
x <- matrix(rnorm(n * p), n, p)
y <- sin(x[, 1]) * sin(x[, 2]) + rnorm(n, mean = 0, sd = 0.2)

# fit HAL model and squash resulting object to reduce footprint
hal_fit <- fit_hal(X = x, Y = y, yolo = FALSE)
squashed <- squash_hal_fit(hal_fit)
```

Index