## Package ‘spikeslab’

October 14, 2022

**Version**  1.1.6  
**Date**  2022-04-26  
**Title**  Prediction and Variable Selection Using Spike and Slab Regression  
**Author**  Hemant Ishwaran <hemant.ishwaran@gmail.com>  
**Maintainer**  Udaya B. Kogalur <ubk@kogalur.com>  
**Depends**  R (>= 3.6.0),  
**Imports**  lars, randomForest, parallel  
**Description**  Spike and slab for prediction and variable selection in linear regression models. Uses a generalized elastic net for variable selection.  
**License**  GPL (>= 3)  
**URL**  https://ishwaran.org/  
**NeedsCompilation**  yes  
**Repository**  CRAN  
**Date/Publication**  2022-04-26 20:50:03 UTC

### R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cv.spikeslab</td>
<td>2</td>
</tr>
<tr>
<td>diabetesI</td>
<td>4</td>
</tr>
<tr>
<td>housingI</td>
<td>5</td>
</tr>
<tr>
<td>leukemia</td>
<td>5</td>
</tr>
<tr>
<td>ozoneI</td>
<td>6</td>
</tr>
<tr>
<td>plot.spikeslab</td>
<td>6</td>
</tr>
<tr>
<td>predict.spikeslab</td>
<td>7</td>
</tr>
<tr>
<td>print.spikeslab</td>
<td>9</td>
</tr>
<tr>
<td>sparsePC.spikeslab</td>
<td>10</td>
</tr>
<tr>
<td>spikeslab</td>
<td>12</td>
</tr>
<tr>
<td>spikeslab.news</td>
<td>16</td>
</tr>
</tbody>
</table>

**Index**  18
cv.spikeslab

K-fold Cross-Validation for Spike and Slab Regression

Description
Computes the K-fold cross-validated mean squared prediction error for the generalized elastic net from spike and slab regression. Returns a stability index for each variable.

Usage
cv.spikeslab(x = NULL, y = NULL, K = 10,
plot.it = TRUE, n.iter1 = 500, n.iter2 = 500, mse = TRUE,
bigp.smalln = FALSE, bigp.smalln.factor = 1, screen = (bigp.smalln),
r.effects = NULL, max.var = 500, center = TRUE, intercept = TRUE,
fast = TRUE, beta.blocks = 5, verbose = TRUE, save.all = TRUE,
ntree = 300, seed = NULL, ...)

Arguments
x x-predictor matrix.
y y-response values.
K Number of folds.
plot.it If TRUE, plots the mean prediction error and its standard error.
n.iter1 Number of burn-in Gibbs sampled values (i.e., discarded values).
n.iter2 Number of Gibbs sampled values, following burn-in.
mse If TRUE, an external estimate for the overall variance is calculated.
bigp.smalln Use if \( p \gg n \).
bigp.smalln.factor Top \( n \) times this value of variables to be kept in the filtering step (used when \( p \gg n \)).
screen If TRUE, variables are first pre-filtered.
r.effects List used for grouping variables (see details below).
max.var Maximum number of variables allowed in the final model.
center If TRUE, variables are centered by their means. Default is TRUE and should only be adjusted in extreme examples.
intercept If TRUE, an intercept is included in the model, otherwise no intercept is included. Default is TRUE.
fast If TRUE, use blocked Gibbs sampling to accelerate the algorithm.
beta.blocks Update beta using this number of blocks (fast must be TRUE).
verbose If TRUE, verbose output is sent to the terminal.
save.all If TRUE, spikeslab object for each fold is saved and returned.
ntree Number of trees used by random forests (applies only when mse is TRUE).
seed Seed for random number generator. Must be a negative integer.
... Further arguments passed to or from other methods.
The `cv.spikeslab` function invisibly returns a list with components:

- `spikeslab.obj`: Spike and slab object from the full data.
- `cv.spikeslab.obj`: List containing spike and slab objects from each fold. Can be NULL.
- `cv.fold`: List containing the cv splits.
- `cv`: Mean-squared error for each fold for the gnet.
- `cv.path`: A matrix of mean-squared errors for the gnet solution path. Rows correspond to model sizes, columns are the folds.
- `stability`: Matrix containing stability for each variable defined as the percentage of times a variable is identified over the K-folds. Also includes bma and gnet coefficient values and their cv-fold-averaged values.
- `bma`: bma coefficients from the full data in terms of the standardized x.
- `bma.scale`: bma coefficients from the full data, scaled in terms of the original x.
- `gnet`: cv-optimized gnet in terms of the standardized x.
- `gnet.scale`: cv-optimized gnet in terms of the original x.
- `gnet.model`: List of models selected by gnet over the K-folds.
- `gnet.path`: gnet path from the full data, scaled in terms of the original x.
- `gnet.obj`: gnet object from fitting the full data (a lars-type object).
- `gnet.obj.vars`: Variables (in order) used to calculate the gnet object.
- `verbose`: Verbose details (used for printing).

The function `cv.spikeslab` is used to perform cross-validation for the spike and slab regression model. It takes several arguments, including the data, formula, and standardization options.

### Author(s)

Hemant Ishwaran (<hemant.ishwaran@gmail.com>)

J. Sunil Rao (<rao.jsunil@gmail.com>)

Udaya B. Kogalur (<ubk@kogalur.com>)

### References


### See Also

sparsePC.spikeslab, plot.spikeslab, predict.spikeslab, print.spikeslab.
Examples

```r
## Not run:
#------------------------------------------------------------
# Example 1: 10-fold validation using parallel processing
#------------------------------------------------------------

data(ozoneI, package = "spikeslab")
y <- ozoneI[, 1]
x <- ozoneI[, -1]
cv.obj <- cv.spikeslab(x = x, y = y, parallel = 4)
plot(cv.obj, plot.type = "cv")
plot(cv.obj, plot.type = "path")

#------------------------------------------------------------
# Example 2: 10-fold validation using parallel processing
# (high dimensional diabetes data)
#------------------------------------------------------------

# add 2000 noise variables
data(diabetesI, package = "spikeslab")
diabetes.noise <- cbind(diabetesI,
noise = matrix(rnorm(nrow(diabetesI) * 2000), nrow(diabetesI)))
x <- diabetes.noise[, -1]
y <- diabetes.noise[, 1]
cv.obj <- cv.spikeslab(x = x, y = y, bigp.smalln=TRUE, parallel = 4)
plot(cv.obj)

## End(Not run)
```

diabetesI

Diabetes Data with Interactions

Description

The data consists of 442 patients in which the response of interest is a quantitative measure of disease progression. The data includes 10 baseline measurements for each patient, in addition to 45 interactions and 9 quadratic terms, for a total of 64 variables for each patient. The outcome is $Y$.

Source


Examples

```r
data(diabetesI, package = "spikeslab")
```
**housingI**

*Boston Housing Interaction Data*

**Description**

Median house price for 506 census tracts of Boston from the 1970 census. The original data comprises 506 observations and 13 variables but has been modified here to include all pairwise interactions of main effects and to include B-spline basis functions of up to 6 degrees of freedom for all original predictors. In addition, all real valued variables were mapped to dummy variables representing a factor with three levels and all pairwise interactions of these dummy variables were added to the design matrix. In total, the modified data contains 506 observations and 658 variables. The outcome is the median house price `medv`.

**Source**


**Examples**

data(housingI, package = "spikeslab")

**leukemia**

*Golub Leukemia Gene Expression Data*

**Description**

Gene expression cancer data set (Golub et al.) of samples from human acute myeloid (AML) and acute lymphoblastic leukemias (ALL). 3571 expression values on 72 individuals.

**Source**


**Examples**

data(leukemia, package = "spikeslab")
**ozoneI**  
*Ozone Interaction Data*

**Description**

The data consists of 366 readings of maximum daily ozone measured in the Los Angeles basin. After removing missing values, the original data has been expanded to include all pairwise interactions, as well as B-spline basis functions (6 degrees of freedom), for each of the original 12 variables (9 meteorological variables and 3 variables recording date of measurement: month, day of the month, and day of week). In total, the modified data has 203 observations and 134 variables. The outcome is ozone.

**Source**


**Examples**

```r
data(ozoneI, package = "spikeslab")
```

**plot.spikeslab**  
*Plots for Spike and Slab Analysis*

**Description**

Plots either the gnet solution path or the cross-validated mean-squared-error (the latter only applies when cross-validation is used).

**Usage**

```r
## S3 method for class 'spikeslab'
plot(x, plot.type = c("path", "cv"), breaks = FALSE, ...)
```

**Arguments**

- `x` An object of class spikeslab.
- `plot.type` Choosing "path" produces a plot of the gnet solution path. The choice "cv" produces the mean-squared error plot. The latter applies only to objects from a cv.spikeslab call.
- `breaks` If TRUE, then vertical lines are drawn at each break point in the gnet solution path.
- `...` Further arguments passed to or from other methods.
**predict.spikeslab**  

**Spike and Slab Prediction**

**Description**

Prediction on test data using spike and slab regression.

**Usage**

```r
## S3 method for class 'spikeslab'
predict(object, newdata = NULL, ...)
```

**Arguments**

- `object`  
  An object of class `spikeslab`.

- `newdata`  
  Data frame or x-matrix containing test data (if omitted, the training data is used).

- `...`  
  Further arguments passed to or from other methods.

**Examples**

```r
## Not run:
#------------------------------------------------------------
# Example 1: diabetes data
#------------------------------------------------------------
data(diabetesI, package = "spikeslab")
obj <- spikeslab(Y ~ ., diabetesI, verbose = TRUE)
plot(obj, plot.type = "path")
## End(Not run)
```

**References**


**See Also**

`spikeslab`, `cv.spikeslab`.
Details

Computes the predicted value using a test data set.

Value

A vector of fitted values for the BMA and gnet and a matrix of fitted values for the gnet path. Also returns the grr mixing predictor if the object has been parsed by the mixing wrapper.

Author(s)

Hemant Ishwaran (<hemant.ishwaran@gmail.com>)
J. Sunil Rao (<rao.jsunil@gmail.com>)
Udaya B. Kogalur (<ubk@kogalur.com>)

References


See Also

spikeslab.

Examples

```r
## Not run:
#------------------------------------------------------------
# Example 1: get the predictor for the training data
#------------------------------------------------------------
data(diabetesI, package = "spikeslab")
x <- diabetesI[, -1]
y <- diabetesI[, 1]
obj <- spikeslab(x = x, y = y)
#gnet predictor
yhat.gnet <- predict(obj)$yhat.gnet
#an equivalent call is...
yhat.gnet <- predict(obj, x = x)$yhat.gnet
```
# Example 2: ozone data with interactions

data(ozoneI, package = "spikeslab")
train.pt <- sample(1:nrow(ozoneI), nrow(ozoneI) * 0.80)
obj <- spikeslab(ozone ~ ., ozoneI[train.pt, ])
ytest <- ozoneI$ozone[-train.pt]
ss.pred <- predict(obj, ozoneI[-train.pt, ])
yhat.bma <- ss.pred$yhat.bma
yhat.gnet <- ss.pred$yhat.gnet
plot(ytest, yhat.bma, ylab = "yhat", pch = 16, col = 4)
points(ytest, yhat.gnet, pch = 16, col = 2)
abline(0, 1, lty = 2, col = 2)
legend("bottomright", legend = c("bma", "gnet"), col = c(4, 2), pch = 16)
## End(Not run)

print.spikeslab

---

Description

Print summary output from spike and slab analysis. Note that this is the default print method for the package.

Usage

```r
## S3 method for class 'spikeslab'
print(x, ...)
```

Arguments

- `x`: An object of class spikeslab.
- `...`: Further arguments passed to or from other methods.

Author(s)

- Hemant Ishwaran (<hemant.ishwaran@gmail.com>)
- J. Sunil Rao (<rao.jsunil@gmail.com>)
- Udaya B. Kogalur (<ubk@kogalur.com>)

References


**See Also**

`spikeslab`.

**Examples**

```r
## Not run:
#------------------------------------------------------------
# Example 1: diabetes data
#------------------------------------------------------------

data(diabetesI, package = "spikeslab")
obj <- spikeslab(Y ~ ., diabetesI, verbose = TRUE)
print(obj)

## End(Not run)
```

**sparePC.spikeslab**  
*Multiclass Prediction using Spike and Slab Regression*

**Description**

Variable selection for the multiclass gene prediction problem.

**Usage**

```r
sparePC.spikeslab(x = NULL, y = NULL, n.rep = 10,
                  n.iter1 = 150, n.iter2 = 100, n.prcmp = 5, max.genes = 100,
                  ntree = 1000, nodesize = 1, verbose = TRUE, ...)
```

**Arguments**

- `x`  
x matrix of gene expressions.
- `y`  
Class labels.
- `n.rep`  
Number of Monte Carlo replicates.
- `n.iter1`  
Number of burn-in Gibbs sampled values (i.e., discarded values).
- `n.iter2`  
Number of Gibbs sampled values, following burn-in.
- `n.prcmp`  
Number of principal components.
- `max.genes`  
Maximum number of genes in final model.
- `ntree`  
Number of trees used by random forests.
- `nodesize`  
Nodesize of trees.
- `verbose`  
If TRUE, verbose output is sent to the terminal.
- `...`  
Further arguments passed to or from other methods.
Details

Multiclass prediction using a hybrid combination of spike and slab linear regression and random
forest multiclass prediction (Ishwaran and Rao, 2009). A pseudo y-vector of response values is
calculated using each of the top \textit{n.prcmp} principal components of the x-gene expression matrix.
The generalized elastic net obtained from using spike and slab regression is used to select genes;
one regression fit is used for each of the pseudo y-response vectors. The final combined set of genes
are passed to random forests and used to construct a multiclass forest predictor. This procedure is
repeated \textit{n.rep} times with each Monte Carlo replicate based on balanced cross-validation with
2/3rds of the data used for training and 1/3rd used for testing.

\textbf{Miscellanea:}
Test set error is only computed when \textit{n.rep} is larger than 1. If \textit{n.rep}=1 the full data is used without
any cross-validation.

Value

Invisibly, the final set of selected genes as well as the complete set of genes selected over the \textit{n.rep}
Monte Carlo replications. The random forest classifier is also returned.

The misclassification error rate, error rate for each class, and other summary information are output
to the terminal.

Author(s)

Hemant Ishwaran (<hemant.ishwaran@gmail.com>)
J. Sunil Rao (<rao.jsunil@gmail.com>)
Udaya B. Kogalur (<ubk@kogalur.com>)

References

Ishwaran H. and Rao J.S. (2009). Generalized ridge regression: geometry and computational solu-
tions when \(p\) is larger than \(n\).

See Also

spikeslab.

Examples

```
## Not run:
#------------------------------------------------------------
# Example 1: leukemia data
#------------------------------------------------------------

data(leukemia, package = "spikeslab")
sparsePC.out <- sparsePC(x = leukemia[, -1], y = leukemia[, 1], n.rep = 3)
rf.obj <- sparsePC.out$rf.obj
varImpPlot(rf.obj)
```

## End(Not run)
spikeslab  

Spike and Slab Regression

Description

Fits a rescaled spike and slab model using a continuous bimodal prior. A generalized elastic net estimator is used for variable selection and estimation. Can be used for prediction and variable selection in low- and high-dimensional linear regression models.

Usage

spikeslab(formula, data = NULL, x = NULL, y = NULL, 
n.iter1 = 500, n.iter2 = 500, mse = TRUE, 
bigp.smalln = FALSE, bigp.smalln.factor = 1, screen = (bigp.smalln), 
r.effects = NULL, max.var = 500, center = TRUE, intercept = TRUE, 
fast = TRUE, beta.blocks = 5, verbose = FALSE, ntree = 300, 
seed = NULL, ...)

Arguments

- formula: A symbolic description of the model to be fit.
- data: Data frame containing the data used in the formula.
- x: x predictor matrix (can be used in place of formula and data frame call).
- y: y response (can be used in place of formula and data frame call).
- n.iter1: Number of burn-in Gibbs sampled values (i.e., discarded values).
- n.iter2: Number of Gibbs sampled values, following burn-in.
- mse: If TRUE, an external estimate for the overall variance is calculated using ridge regression or random forests (the latter is used when the degrees of freedom are low). Otherwise, the variance is included in the prior and estimated using Gibbs sampling.
- bigp.smalln: Use if p > n.
- bigp.smalln.factor: Removes all variables except the top n times bigp.smalln.factor ones (used in filtering when p > n).
- screen: If TRUE, variables are pre-filtered.
- r.effects: List used for grouping variables (see details below).
- max.var: Maximum number of variables allowed in the final model.
- center: If TRUE, variables are centered by their means. Default is TRUE and should only be adjusted in extreme examples.
- intercept: If TRUE, an intercept is included in the model, otherwise no intercept is included. Default is TRUE.
- fast: If TRUE, use blocked Gibbs sampling to accelerate the algorithm.
beta.blocks  Update beta using this number of blocks (fast must be TRUE).
verbose      If TRUE, verbose output is sent to the terminal.
ntree        Number of trees used by random forests (applies only when mse is TRUE).
seed         Seed for random number generator. Must be a negative integer.
...          Further arguments passed to or from other methods.

Details

—> General:

The spike and slab method is described in detail in Ishwaran and Rao (2003, 2005a, 2005b and 2009). For high-dimensional problems in which \( p \gg n \), where \( p \) is the number of variables and \( n \) is the sample size, use the option bigp. smalln=TRUE. Doing so implements a three-stage procedure:

(1) Filtering step. This removes all variables except the top \( n \) times bigp. smalln.factor ones. Uses spike and slab regression with grouped regularization (complexity) parameters.

(2) Model averaging step. Refit the model using only those predictors from step 1. Returns the posterior mean values from fitting a spike and slab model; referred to as the Bayesian model averaged (bma) estimate.

(3) Variable selection step. Select variables using the generalized elastic net (gnet).

The filtering step is omitted when bigp. smalln=FALSE. Filtering can however be requested by setting screen=TRUE although users should be aware that this may degrade performance and should only be used when \( p \) is on the same order of \( n \).

Variables can be grouped using r.effects. Grouping has the effect of forcing variables within a given group to share a common complexity (regularization) parameter. To do so, define a list with each entry in the list made up of the variable names to be grouped. There is no limit to the number of groups. Any variable that does not appear in the list will be assigned to a default group (the default group also has its own group-specific regularization parameter). See Examples 1 and 3 below.

—> Miscellanea:

By default, fast=TRUE when bigp. smalln=TRUE. This invokes an ultra-fast filtering step. Setting fast=FALSE invokes a more thorough filtering method that may slightly improve inferential results, but computational times will become very slow. The trade-off is unlikely to be justified.

The formula and data-frame call should be avoided in high-dimensional problems and instead the x-predictor matrix and y response vector should be passed directly (see Example 3). This avoids the huge overhead in parsing formula in R.

By default, predictors are normalized to have mean 0 and variance 1. Pre-processing also involves centering y unless the user specifically requests that the intercept be excluded from the model. Users can also over-ride centering predictors by setting center=FALSE. Use with extreme care.

The verbose option sends output to the terminal showing the number of Gibbs iterations and the current complexity (regularization) parameter(s).

Depends on the randomForest package for estimating the variance when mse=TRUE. Note that mse is over-ridden and set to FALSE when bigp. smalln=TRUE.

Depends on the lars package for the variable slection step.
Value

An object of class spikeslab with the following components:

- `summary` Summary object.
- `verbose` Verbose details (used for printing).
- `terms` Terms.
- `sigma.hat` Estimated variance.
- `y` Original y.
- `xnew` Centered, rescaled x-matrix.
- `x` Original x-matrix.
- `y.center` Centering for original y.
- `x.center` Centering for original x-matrix.
- `x.scale` Scaling for original x-matrix.
- `names` Variable names.
- `bma` bma coefficients in terms of xnew.
- `bma.scale` bma coefficients rescaled in terms of original x.
- `gnet` gnet coefficients in terms of xnew.
- `gnet.scale` gnet coefficients rescaled in terms of original x.
- `gnet.path` gnet path scaled in terms of the original x.
- `gnet.obj` gnet object (a lars-type object).
- `gnet.obj.vars` Variables (in order) used to calculate the gnet object.
- `gnet.parms` Generalized ridge regression parameters used to define the gnet.
- `phat` Estimated model dimension.
- `complexity` Complexity (regularization) parameter estimates.
- `ridge` List containing ridge values used to determine the bma.
- `models` List containing the models sampled.

Author(s)

Hemant Ishwaran (<hemant.ishwaran@gmail.com>)
J. Sunil Rao (<rao.jsunil@gmail.com>)
Udaya B. Kogalur (<ubk@kogalur.com>)

References


**See Also**

cv.spikeslab, plot.spikeslab, predict.spikeslab, print.spikeslab, sparsePC.spikeslab.

**Examples**

```r
# Example 1: diabetes data
#------------------------------------------------------------
# basic call
data(diabetesI, package = "spikeslab")
obj <- spikeslab(Y ~ . , diabetesI, verbose=TRUE)
print(obj)
plot(obj)

# grouping effect
# separate main effects and interactions into two groups
# use a group-specific regularization parameter for each group
xnames <- names(diabetesI[, -1])
r.eff <- vector("list", 2)
r.eff[[1]] <- xnames[1:10]
r.eff[[2]] <- xnames[-c(1:10)]
obj2 <- spikeslab(Y ~ . , diabetesI, verbose=TRUE, r.effects=r.eff)
obj2
# extract the regularization parameters
print(apply(obj2$complexity, 2, summary))

# Example 2: high-dimensional noise (diabetes data)
#------------------------------------------------------------
# add 2000 noise variables
data(diabetesI, package = "spikeslab")
diabetes.noise <- cbind(diabetesI,
        noise = matrix(rnorm(nrow(diabetesI) * 2000), nrow(diabetesI)))
```
# example of a big p, small n call
# don't use formula call; make call with x and y arguments
x <- diabetes.noise[, -1]
y <- diabetes.noise[, 1]
obj <- spikeslab(x=x, y=y, verbose=TRUE, bigp.smalln=TRUE, max.var=100)
obj

# same example ... but now group variables
r.eff <- vector("list", 2)
r.eff[[1]] <- names(x)[c(1:100)]
r.eff[[2]] <- names(x)[-c(1:100)]
obj2 <- spikeslab(x=x, y=y, verbose=TRUE, bigp.smalln=TRUE,
                   r.effects=r.eff, max.var=100)
obj2

#------------------------------------------------------------
# Example 3: housing data with interactions
#------------------------------------------------------------

# another example of a big p, small n call
data(housingI, package = "spikeslab")
obj <- spikeslab(medv ~ ., housingI, verbose = TRUE,
                 bigp.smalln = TRUE, max.var = 200)
print(obj)

## End(Not run)

spikeslab.news

---

**Description**

Show the NEWS file of the *spikeslab* package.

**Usage**

spikeslab.news(...)

**Arguments**

... Further arguments passed to or from other methods.

**Value**

None.
Author(s)

Hemant Ishwaran <hemant.ishwaran@gmail.com>
Index

* datasets
  diabetesI, 4
  housingI, 5
  leukemia, 5
  ozoneI, 6

* documentation
  spikeslab.news, 16

* plot
  plot.spikeslab, 6

* print
  print.spikeslab, 9

* regression
  cv.spikeslab, 2
  predict.spikeslab, 7
  sparsePC.spikeslab, 10
  spikeslab, 12
  cv.spikeslab, 2
  diabetesI, 4
  housingI, 5
  leukemia, 5
  ozoneI, 6
  plot.spikeslab, 6
  predict.spikeslab, 7
  print.spikeslab, 9
  sparsePC(sparsePC.spikeslab), 10
  sparsePC.spikeslab, 10
  spikeslab, 12
  spikeslab.news, 16