Package ‘L0Learn’

June 21, 2018

Type Package

Title Fast Algorithms for Best Subset Selection

Version 0.3-0

Date 2018-06-20

Author Hussein Hazimeh
Rahul Mazumder

Maintainer Hussein Hazimeh <hazimeh@mit.edu>

Description Highly optimized coordinate descent and local combinatorial search algorithms for (approximately) solving L0-regularized learning problems.

License MIT + file LICENSE

Imports Rcpp (>= 0.12.13), Matrix, methods

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

R topics documented:

coef.L0Learn *Extract Solutions*

Description

Extracts a specific solution in the regularization path

Usage

```
## S3 method for class 'L0Learn'
coef(object, lambda, gamma = 0, ...)

## S3 method for class 'L0LearnCV'
coef(object, lambda, gamma = 0, ...)
```

Arguments

`object` The output of L0Learn.fit or L0Learn.cvfit

`lambda` The value(s) of lambda at which to extract the solution.

`gamma` The value of gamma at which to extract the solution. Note that, unlike lambda, this can only take single values.

`...` ignore

GenSynthetic *Generate Synthetic Data*

Description

Generates a synthetic dataset as follows: 1) Sample every element in data matrix X from N(0,1). 2) Generate a vector B with the first k entries set to 1 and the rest are zeros. 3) Sample every element in the noise vector e from N(0,1). 4) Set y = XB + e.

Usage

```
GenSynthetic(n, p, k, seed)
```

Arguments

`n` Number of samples

`p` Number of features

`k` Number of non-zeros in true vector of coefficients

`seed` The seed used for randomly generating the data

Value

A list containing the data matrix X and the response vector y.

Examples

```
data <- GenSynthetic(n=500,p=1000,k=10,seed=1)
X = data$X
y = data$y
```

L0Learn.cvfit *Cross Validation*

Description

Fits an L0 model on the full data and performs K-fold cross-validation.

Usage

```
L0Learn.cvfit(x, y, loss = "SquaredError", penalty = "L0",
  algorithm = "CD", maxSuppSize = 100, nLambda = 100, nGamma = 10,
  gammaMax = 10, gammaMin = 1e-04, partialSort = TRUE, maxIters = 200,
  tol = 1e-06, activeSet = TRUE, activeSetNum = 3, maxSwaps = 100,
  scaleDownFactor = 0.8, screenSize = 1000, autoLambda = TRUE,
  lambdaGrid = list(0), nFolds = 10, seed = 1)
```

Arguments

x	The data matrix.
y	The response vector.
loss	The loss function to be minimized. The currently supported choice is "SquaredError".
penalty	The type of regularization. This can take either one of the following choices: "L0", "L0L2", and "L0L1".
algorithm	The type of algorithm used to minimize the objective. Currently "CD" and "CDPSI" are are supported. "CD" is a variant of cyclic coordinate descent and can run very fast. "CDPSI" performs local combinatorial search on top of CD and thus can achieve higher quality solutions (at the expense of increased running time).
maxSuppSize	The maximum support size to reach in the grid before termination. We recommend setting this to a small fraction of min(n,p) (e.g. 0.05 * min(n,p)) as L0 regularization typically selects a small portion of non-zeros.
nLambda	The number of Lambda values to select (recall that Lambda is the regularization parameter corresponding to the L0 norm).
nGamma	The number of Gamma values to select (recall that Gamma is the regularization parameter corresponding to L1 or L2, depending on the chosen penalty).

`gammaMax` The maximum value of Gamma when using the L0L2 penalty. For the L0L1 penalty this is automatically selected by the toolkit.

`gammaMin` The minimum value of Gamma when using the L0L2 penalty. For the L0L1 penalty, gammaMin specifies the fraction of gammaMax at which the grid ends.

`partialSort` If TRUE partial sorting will be used for sorting the coordinates (see our paper for for details). Otherwise, full sorting is used.

`maxIters` The maximum number of iterations (full cycles) for CD per grid point.

`tol` The tolerance which decides when to terminate CD (based on the relative change in the objective).

`activeSet` If TRUE, performs active set updates.

`activeSetNum` The number of consecutive times a support should appear before declaring support stabilization.

`maxSwaps` The maximum number of swaps used by CDPSI for each grid point.

`scaleDownFactor`
This parameter decides how close the selected Lambda values are. The choice should be between strictly between 0 and 1 (i.e., 0 and 1 are not allowed). For details, see our paper - Section 5 on Adaptive Selection of Tuning Parameters).

`screenSize` The number of coordinates to cycle over when performing correlation screening.

`autoLambda` If FALSE, the user specifier a grid of Lambda0 values through the Lambda0Grid parameter. Otherwise, if TRUE, the values of Lambda0 are automatically selected based on the data.

`lambdaGrid` A vector of Lambda0 values to use in computing the regularization path. This is ignored unless autoLambda0 = FALSE.

`nFolds` The number of folds for cross-validation.

`seed` The seed used in randomly shuffling the data for cross-validation.

Value

An S3 object of type "L0Learn" describing the regularization path. The object has the following members.

`cvMeans` For L0, this is a sequence of cross-validation errors: cvMeans[i] corresponds to the solution indexed by lambda[i]. For L0L1 and L0L2, cvMeans is a list, where each element is a sequence corresponding to a particular gamma, i.e., cvMeans[[i]] is the sequence of cross-validation errors corresponding to gamma[i].

`cvSDs` For L0, this is a sequence of standard deviations for the cross-validation errors. For L0L1 and L0L2, it is a list of sequences: cvSDs[[i]] corresponds to cvMeans[[i]].

`fit` The fitted model with type "L0Learn", i.e., this is the same object returned by `L0Learn.fit`.

L0Learn.fit *Fit an L0-regularized model*

Description

Computes the regularization path for the specified loss function and choice of regularization (which can be a combination of the L0, L1, and L2 (squared) norms).

Usage

```
L0Learn.fit(x, y, loss = "SquaredError", penalty = "L0", algorithm = "CD",
  maxSuppSize = 100, nLambda = 100, nGamma = 10, gammaMax = 10,
  gammaMin = 1e-04, partialSort = TRUE, maxIters = 200, tol = 1e-06,
  activeSet = TRUE, activeSetNum = 3, maxSwaps = 100,
  scaleDownFactor = 0.8, screenSize = 1000, autoLambda = TRUE,
  lambdaGrid = list(0))
```

Arguments

`x`	The data matrix.
`y`	The response vector.
`loss`	The loss function to be minimized. The currently supported choice is "SquaredError".
`penalty`	The type of regularization. This can take either one of the following choices: "L0", "L0L2", and "L0L1".
`algorithm`	The type of algorithm used to minimize the objective. Currently "CD" and "CDPSI" are are supported. "CD" is a variant of cyclic coordinate descent and can run very fast. "CDPSI" performs local combinatorial search on top of CD and thus can achieve higher quality solutions (at the expense of increased running time).
`maxSuppSize`	The maximum support size to reach in the grid before termination. We recommend setting this to a small fraction of min(n,p) (e.g. 0.05 * min(n,p)) as L0 regularization typically selects a small portion of non-zeros.
`nLambda`	The number of Lambda values to select (recall that Lambda is the regularization parameter corresponding to the L0 norm).
`nGamma`	The number of Gamma values to select (recall that Gamma is the regularization parameter corresponding to L1 or L2, depending on the chosen penalty).
`gammaMax`	The maximum value of Gamma when using the L0L2 penalty. For the L0L1 penalty this is automatically selected by the toolkit.
`gammaMin`	The minimum value of Gamma when using the L0L2 penalty. For the L0L1 penalty, gammaMin specifies the fraction of gammaMax at which the grid ends.
`partialSort`	If TRUE partial sorting will be used for sorting the coordinates (see our paper for for details). Otherwise, full sorting is used.
`maxIters`	The maximum number of iterations (full cycles) for CD per grid point.

`tol` The tolerance which decides when to terminate CD (based on the relative change in the objective).

`activeSet` If TRUE, performs active set updates.

`activeSetNum` The number of consecutive times a support should appear before declaring support stabilization.

`maxSwaps` The maximum number of swaps used by CDPSI for each grid point.

`scaleDownFactor` This parameter decides how close the selected Lambda values are. The choice should be between strictly between 0 and 1 (i.e., 0 and 1 are not allowed). For details, see our paper - Section 5 on Adaptive Selection of Tuning Parameters).

`screenSize` The number of coordinates to cycle over when performing correlation screening.

`autoLambda` If FALSE, the user specifier a grid of Lambda0 values through the Lambda0Grid parameter. Otherwise, if TRUE, the values of Lambda0 are automatically selected based on the data.

`lambdaGrid` A vector of Lambda0 values to use in computing the regularization path. This is ignored unless autoLambda0 = FALSE.

Value

An S3 object of type "L0Learn" describing the regularization path. The object has the following members.

`a0` For L0, this is a sequence of intercepts. Note for L0L1 and L0L2, a0 is a list of intercept sequences, where each member of the list corresponds to a single gamma value.

`beta` For L0, this is a matrix of coefficients of dimensions p x `length(lambda)`, where each column corresponds to a single lambda value. For L0L1 and L0L2, this is a list of coefficient matrices, where each matrix corresponds to a single gamma value.

`lambda` For L0, lambda is a sequence of lambda values. For L0L1 and L0L1, it is a list of lambda sequences, each corresponding to a single gamma value.

`gamma` For L0L1 and L0L2, this is a sequence of gamma values.

`suppSize` For L0, this is a sequence of support sizes (number of non-zero coefficients). For L0L1 and L0L2, it is a list of support size sequences, each representing a single gamma value.

`converged` For L0, this is a sequence indicating whether the algorithm converged at the current point in the regularization path. For L0L1 and L0L2, this is a list of sequences, each representing a single gamma value.

plot.L0LearnCV *Plot Cross-validation Errors*

Description

Plots cross-validation errors

Usage

```
## S3 method for class 'L0LearnCV'
plot(x, gamma, ...)
```

Arguments

`x`	L0Learn.fit object
`gamma`	The gamma value for L0L1 and L0L2 models. This is ignored for L0.
`...`	ignore

predict.L0Learn *Predict Response*

Description

Predicts the response for a given sample

Usage

```
## S3 method for class 'L0Learn'
predict(object, newx, lambda, gamma = 0, ...)

## S3 method for class 'L0LearnCV'
predict(object, newx, lambda, gamma = 0, ...)
```

Arguments

`object`	The output of L0Learn.fit or L0Learn.cvfit
`newx`	A matrix on which predictions are made. The matrix should have p columns.
`lambda`	The value(s) of lambda to use for prediction. A summary of the lambdas in the regularization path can be obtained using `print(fit)`.
`gamma`	The value of gamma to use for prediction. A summary of the gammas in the regularization path can be obtained using `print(fit)`.
`...`	ignore

print.L0Learn *Print L0Learn.fit object*

Description

Prints a summary of L0Learn.fit

Usage

```
## S3 method for class 'L0Learn'
print(x, ...)

## S3 method for class 'L0LearnCV'
print(x, ...)
```

Arguments

x	The output of L0Learn.fit or L0Learn.cvfit
...	ignore

Index