Package ‘casecontrolSL’

February 7, 2014

Type Package

Title Case-Control Subsampling for SuperLearner

Version 0.1-5

Date 2014-02-07

Author Erin LeDell, Mark van der Laan, Maya Petersen

Maintainer Erin LeDell <ledell@berkeley.edu>

Description This package is an add-on to the base SuperLearner package. It implements a case-control subsampling of the training data using inverse probability censored weighting, for cross-validated SuperLearner.

License GPL (>=2)

Depends R (>= 2.14.0), nnls, SuperLearner

Suggests arm, BayesTree, caret, class, e1071, earth, gam, gbm, ggplot2, glmnet, Hmisc, ipred, lattice, MASS, mda, mlbench, nnet, parallel, party, polspline, quadprog, randomForest, rpart, SIS, spls, stepPlr

LazyLoad yes

R topics documented:

  casecontrolSL-package ........................................... 2
  casecontrolSL ..................................................... 3
  CV.casecontrolSL ................................................ 5
  CVFolds2 .......................................................... 7
  CVSubfolds ......................................................... 8
  predict.casecontrolSL ......................................... 9

Index 10
casecontrolSL-package  Cross-validated Case-Control Subsampling for Super Learner

Description

This package is an extension to the SuperLearner package. It implements a case-control subsampling of the training data using inverse probability of censoring weighting (IPCW), for cross-validated SuperLearner.

Details

| Package: | casecontrolSL |
| Type:    | Package       |
| Version: | 1.0-5         |
| Date:    | 2013-02-07    |
| License: | GPL>=2        |

The main functions of this package are `casecontrolSL` and `CV.casecontrolSL`. These functions take identical arguments as their SuperLearner counterparts, with the addition of an extra variable, `subsize`, used for subset selection. The `subsize` variable specifies size of the subset of the data that will be used to cross-validate SuperLearner. By default, all cases will be included in the subset and the remaining data points will be controls that are sampled randomly.

Author(s)

Erin LeDell, Mark van der Laan, Maya Petersen

Maintainer: Erin LeDell <ledell@berkeley.edu>

References


See Also

SuperLearner

Examples

### To be updated.
Description

A prediction function for Case-Control Super Learner. The casecontrolSL function takes a training set pair (X,Y) and returns the predicted values based on a validation set. However, the training is performed using a case-control sub-sampled subset of the data, the size of which is specified by the subsize argument.

Usage

casecontrolSL(Y, X, newX = NULL, SL.library, method = "method.NNLS", id = NULL, verbose = FALSE, control = list(), cvControl = list(stratifyCV=TRUE), obsWeights = NULL, subsize = 2*length(which(Y==1)))

Arguments

Y
The outcome in the training data set. Must be a numeric vector.

X
The predictor variables in the training data set, usually a data.frame.

newX
The predictor variables in the validation data set. The structure should match X. If missing, uses X for newX.

SL.library
Either a character vector of prediction algorithms or a list containing character vectors. See details below for examples on the structure. A list of functions included in the SuperLearner package can be found with listWrappers().

method
A list (or a function to create a list) containing details on estimating the coefficients for the super learner and the model to combine the individual algorithms in the library. See ?method.template for details. Currently, the built in options are either "method.NNLS" (the default), "method.NNLS2", or "method.NNloglik". NNLS and NNLS2 are non-negative least squares based on the Lawson-Hanson algorithm and the dual method of Goldfarb and Idnani, respectively. NNloglik is a non-negative binomial likelihood maximization using the BFGS quasi-Newton optimization method.

id
Optional cluster identification variable. For the cross-validation splits, id forces observations in the same cluster to be in the same validation fold. id is passed to the prediction and screening algorithms in SL.library, but be sure to check the individual wrappers as many of them ignore the information.

verbose
Logical; TRUE for printing progress during the computation (helpful for debugging).

control
A list of parameters to control the estimation process. Parameters include saveFitLibrary and trimLogit. See SuperLearner.control for details.

cvControl
A list of parameters to control the cross-validation process. Parameters include V, stratifyCV, shuffle and validRows. See SuperLearner.CV.control for details.
Optional observation weights variable. As with `id` above, `obsWeights` is passed to the prediction and screening algorithms, but many of the built-in wrappers ignore (or can’t use) the information. If you are using observation weights, make sure the library you specify uses the information.

The size of the case-control subset used for training. All of the cases will be automatically included in this subset, so choose a number higher than the number of cases plus one.

After subsampling has taken place, the subsetted training data is passed to the `SuperLearner` function. The weights for each algorithm in `SL.library` is estimated, along with the fit of each algorithm.

An object of class `casecontrolSL` (a list) with components:

- `call`: The matched call.
- `libraryNames`: A character vector with the names of the algorithms in the library. The format is `“predictionAlgorithm_screeningAlgorithm”` with `“_All”` used to denote the prediction algorithm run on all variables in `X`.
- `SL.library`: Returns `SL.library` in the same format as the argument with the same name above.
- `SL.predict`: The predicted values from the super learner for the rows in `newX`.
- `coef`: Coefficients for the super learner.
- `library.predict`: A matrix with the predicted values from each algorithm in `SL.library` for the rows in `newX`.
- `Z`: The Z matrix (the cross-validated predicted values for each algorithm in `SL.library`).
- `cvRisk`: A numeric vector with the V-fold cross-validated risk estimate for each algorithm in `SL.library`. Note that this does not contain the CV risk estimate for the SuperLearner, only the individual algorithms in the library.
- `family`: Returns the `family` value from above. This will always be ‘binomial’.
- `fitLibrary`: A list with the fitted objects for each algorithm in `SL.library` on the full training data set.
- `varNames`: A character vector with the names of the variables in `X`.
- `validRows`: A list containing the row numbers for the V-fold cross-validation step.
- `method`: A list with the method functions.
- `whichScreen`: A logical matrix indicating which variables passed each screening algorithm.
- `control`: The `control` list.
- `cvControl`: The `cvControl` list.
- `errorsInCVLibrary`: A logical vector indicating if any algorithms experienced an error within the CV step.
- `errorsInLibrary`: A logical vector indicating if any algorithms experienced an error on the full data.
CV.casecontrolSL

subsize  The size of the case-control subset used for training, as specified.

subfolds  A list of indices in the original X, Y that were selected into the training subset, divided into internal cross-validation folds. The total number of indices will equal the subsize value.

Author(s)

Erin LeDell <ledell@berkeley.edu>

See Also

CV.casecontrolSL, SuperLearner

Examples

# Will be added at a later date.

CV.casecontrolSL

Function to get V-fold cross-validated risk estimate for Super Learner

Description

Function to get V-fold cross-validated risk estimate for Super Learner. This function splits the data into V folds and then calls SuperLearner. However, as opposed to the CV.SuperLearner function, the training is performed on a case-control sub-sampled subset of the data, but is validated on the original folds. Most of the arguments are passed directly to the SuperLearner function.

Usage

CV.casecontrolSL(Y, X, V = 2/zero.noslash, SL.library, method = "method.NNLS", id = NULL, verbose = FALSE, control = list(saveFitLibrary = FALSE), cvControl = list(), obsWeights = NULL, saveAll = TRUE, parallel = "seq", subsize = max(2*length(which(Y==1)), 2*V )

Arguments

Y  The outcome in the training data set. Must be a numeric vector.
X  The predictor variables in the training data set, usually a data.frame.
V  The number of folds for CV.SuperLearner. This is not the number of folds for SuperLearner. The number of folds for SuperLearner is controlled with cvControl.
SL.library  Either a character vector of prediction algorithms or a list containing character vectors. See details below for examples on the structure. A list of functions included in the SuperLearner package can be found with listWrappers().
method  A list (or a function to create a list) containing details on estimating the coefficients for the super learner and the model to combine the individual algorithms in the library. See ?method.template for details. Currently, the built in options are either "method.NNLS" (the default), "method.NNLS2", or "method.NNloglik". NNLS and NNLS2 are non-negative least squares based on the Lawson-Hanson
CV.casecontrolSL

algorithm and the dual method of Goldfarb and Idnani, respectively. NNLS and NNLS2 will work for both gaussian and binomial outcomes. NNloglik is a non-negative binomial likelihood maximization using the BFGS quasi-Newton optimization method.

id
Optional cluster identification variable. For the cross-validation splits, id forces observations in the same cluster to be in the same validation fold. id is passed to the prediction and screening algorithms in SL.library, but be sure to check the individual wrappers as many of them ignore the information.

verbose
Logical; TRUE for printing progress during the computation (helpful for debugging).

type
A list of parameters to control the estimation process. Parameters include saveFitLibrary and trimLogit. See SuperLearner.control for details.

cvControl
A list of parameters to control the cross-validation process. Parameters include V, stratifyCV, shuffle and validRows. See SuperLearner.CV.control for details.

obsWeights
Optional observation weights variable. As with id above, obsWeights is passed to the prediction and screening algorithms, but many of the built in wrappers ignore (or can’t use) the information. If you are using observation weights, make sure the library you specify uses the information.

saveAll
Logical; Should the entire SuperLearner object be saved for each fold?

parallel
Options for parallel computation of the V-fold step. Use parallel = 'seq' for sequential computation. parallel = 'multicore' to use mclapply for the V-fold step (but note that SuperLearner() will still be sequential). Or parallel can be the name of a snow cluster and will use parLapply for the V-fold step. For both multicore and snow, the inner SuperLearner calls will be sequential.

subsize
The size of the case-control subset used for training. All of the cases will be automatically included in this subset, so choose a number higher than the number of cases plus one.

Details

The SuperLearner function builds a estimator, but does not contain an estimate on the performance of the estimator. Various methods exist for estimator performance evaluation. If you are familiar with the super learner algorithm, it should be no surprise we recommend using cross-validation to evaluate the honest performance of the super learner estimator. The function CV.casecontrolSL computes the usual V-fold cross-validated risk estimate for the super learner (and all algorithms in SL.library for comparison) using a case-control subset of the original data.

Value

An object of class CV.casecontrolSL (a list) with components:

call
The matched call.

AllSL
If saveAll = TRUE, a list with output from each call to SuperLearner, otherwise NULL.

SL.predict
The predicted values from the super learner when each particular row was part of the validation fold.

discreteSL.predict
The traditional cross-validated selector. Picks the algorithm with the smallest cross-validated risk (in super learner terms, gives that algorithm coefficient 1 and all others 0).
whichDiscreteSL
A list of length V. The elements in the list are the algorithm that had the smallest cross-validated risk estimate for that fold.

library.predict
A matrix with the predicted values from each algorithm in SL.library. The columns are the algorithms in SL.library and the rows represent the predicted values when that particular row was in the validation fold (i.e. not used to fit that estimator).

coeff
A matrix with the coefficients for the super learner on each fold. The columns are the algorithms in SL.library the rows are the folds.

cvFolds2(N, id, Y, cvControl)
Generate list of row numbers for each fold in the cross-validation.

Description
Generate list of row numbers for each fold in the cross-validation. CVFolds2 is used in CV.casecontrolSL to create the cross-validation splits. Note: This is an updated version of CVFolds function in the SuperLearner package, which is capable of stratifying the data by response and id simultaneously. This function will be deprecated when CVFolds in SuperLearner is updated to include this capability.

Usage
CVFolds2(N, id, Y, cvControl)
CVSubfolds

Arguments

N Sample size
id Optional cluster id variable. If present, all observations in the same cluster will always be in the same split.
Y outcome
cvControl Control parameters for the cross-validation step. See SuperLearner.CV.control for details.

Value

validRows A list of length V where each element in the list is a vector with the row numbers of the corresponding validation sample.

Author(s)

Erin LeDell <ledell@berkeley.edu>

Description

Generates a case-control subset of the folds, such that the cumulative number of observations across all folds is equal to the subsize argument. This subset will include all cases in each fold and a random draw of the controls.

Usage

CVSubfolds(folds, Y, subsize)

Arguments

folds A folds object returned from CVFolds2.
Y The original outcome vector.
subsize The desired size of the case-control subset. The cumulative number of observations across all folds will be equal to this number.

Value

validRows A list of length V where each element in the list is a vector with the row numbers of the corresponding validation sample.

Author(s)

Erin LeDell <ledell@berkeley.edu>
predict.casecontrolSL  Predict method for casecontrolSL object

Description

Obtains predictions on a new data set from a Case-Control Super Learner (casecontrolSL, fit. May require the original data if one of the library algorithms uses the original data in its predict method.

Usage

## S3 method for class 'casecontrolSL'
predict(object, newdata, family, X = NULL, Y = NULL, ...)

Arguments

object  Fitted object from casecontrolSL
newdata  New X values for prediction
family  Currently allows gaussian or binomial to describe the error distribution
X  Original data set used to fit object
Y  Original outcome used to fit object
...  Additional arguments passed to the predict.SL.* functions

Details

If newdata is omitted the predicted values from object are returned. Each algorithm in the Super Learner library needs to have a corresponding prediction function with the “predict.” prefixed onto the algorithm name (e.g. predict.SL.glm for SL.glm).

Value

pred  Predicted values from Super Learner fit
library.predict  Predicted values for each algorithm in library

Author(s)

Erin LeDell <ledell@berkeley.edu>

See Also

casecontrolSL
Index

*Topic models
  casecontrolSL, 3
  CV.casecontrolSL, 5
  predict.casecontrolSL, 9

*Topic package
  casecontrolSL-package, 2

*Topic utilities
  CVFolds2, 7
  CVSubfolds, 8

casecontrolSL, 3, 9
casecontrolSL-package, 2
CV.casecontrolSL, 5, 5, 7
CV.SuperLearner, 7
CVFolds2, 7
CVSubfolds, 8

predict.casecontrolSL, 9

SuperLearner, 2, 5
SuperLearner.control, 3, 6
SuperLearner.CV.control, 3, 6, 8