Package 'clogitL1'

May 8, 2026

Type Package

Title Fitting Exact Conditional Logistic Regression with Lasso and Elastic Net Penalties

Version 1.5

Date 2019-02-01

Author Stephen Reid and Robert Tibshirani

Maintainer Stephen Reid <sreid1652@gmail.com>

Description Tools for the fitting and cross validation of exact conditional logistic regression models with lasso and elastic net penalties. Uses cyclic coordinate descent and warm starts to compute the entire path efficiently.

License GPL-2

Depends Rcpp (>= 0.10.2)

LinkingTo Rcpp

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-02-02 22:33:36 UTC

Contents

clogitL1-package *Penalised conditional logistic regression.*

Description

Tools for the fitting and cross validation of exact conditional logistic regression models with lasso and elastic net penalties. Uses cyclic coordinate descent and warm starts to compute the entire path efficiently.

Details

Package:	clogitL1
Type:	Package
Version:	1.4
Date:	2013-05-06
License:	GPL-2

Very simple to use. The main fitting function `clogitL1` accepts x, y data and a strata vector indicating stratum membership. It fits the exact conditional logistic regression model at a grid of regularisation parameters. Only 7 functions:

- `clogitL1`
- `cv.clogitL1`
- `plot.clogitL1`
- `plot.cv.clogitL1`
- `print.clogitL1`
- `summary.clogitL1`
- `summary.cv.clogitL1`

Author(s)

Stephen Reid and Rob Tibshirani

Maintainer: Stephen Reid <sreid@stanford.edu>

References

http://www.jstatsoft.org/v58/i12/

clogitL1 *Conditional logistic regression with elastic net penalties*

Description

Fit a sequence of conditional logistic regression models with lasso or elastic net penalties

Usage

```
 clogitL1 (x, y, strata, numLambda=100,
minLambdaRatio=0.000001, switch=0, alpha = 1)
```

Arguments

x	matrix with rows equalling the number of observations. Contains the p-vector regressor values as rows
y	vector of binary responses with 1 for cases and 0 for controls.
strata	vector with stratum membership of each observation.
numLambda	number of different values of the regularisation parameter λ at which to compute parameter estimates. First fit is made at value just below smallest regularisation parameter value at which all parameter estimates are 0; last fit made at this value multipled by minLambdaRatio
minLambdaRatio	ratio of smallest to larget value of regularisation parameter λ at which we find parameter estimates.
switch	index (between 0 and numLambda) at which we transition from linear to logarithmic jumps.
alpha	parameter controling trade off between lasso and ridge penalties. At value 1, we have a pure lasso penalty; at 0, pure ridge. Intermediate values provide a mixture of the two.

Details

The sequence of models implied by numLambda and minLambdaRatio is fit by coordinate descent with warm starts and sequential strong rules. If alpha=1, we fit using a lasso penalty. Otherwise we fit with an elastic net penalty. Note that a pure ridge penalty is never obatined, because the function sets a floor for alpha at 0.000001. This improves the stability of the algorithm. A similar lower bound is set for minLambdaRatio. The sequence of models can be truncated at fewer than numLambda models if it is found that a very large proportion of training set deviance is explained by the model in question.

Value

An object of type clogitL1 with the following fields:

beta	(numLambda + 1)-by-p matrix of estimated coefficients. First row has all 0s

lambda	vector of length numLambda + 1 containing the value of the regularisation parameter at which we obtained the fits.
nz_beta	vector of length numLambda + 1 containing the number of nonzero parameter estimates for the fit at the corresponding regularisation parameter.
ss_beta	vector of length numLambda + 1 containing the number of predictors considered by the sequential strong rule at that iteration.
dev_perc	vector of length numLambda + 1 containing the percentage of null deviance explained by the model represented by that row in the matrix.
y_c	reordered vector of responses. Grouped by stratum with cases coming first.
X_c	reordered matrix of predictors. See above.
strata_c	reordered stratum vector. See above.
nVec	vector of length the number of unique strata in strata containing the number of observations encountered in each stratum.
mVec	vector containing the number of cases in each stratum.
alpha	penalty trade off parameter.

References

http://www.jstatsoft.org/v58/i12/

See Also

plot.clogitL1

Examples

```
set.seed(145)
# data parameters
K = 10 # number of strata
n = 5 # number in strata
m = 2 # cases per stratum
p = 20 # predictors

# generate data
y = rep(c(rep(1, m), rep(0, n-m)), K)
X = matrix (rnorm(K*n*p, 0, 1), ncol = p) # pure noise
strata = sort(rep(1:K, n))

par(mfrow = c(1,2))
# fit the conditional logistic model
clObj = clogitL1(y=y, x=X, strata)
plot(clObj, logX=TRUE)

# cross validation
clcvObj = cv.clogitL1(clObj)
plot(clcvObj)
```

`cv.clogitL1` *Cross validation of conditional logistic regression with elastic net penalties*

Description

Find the best of a sequence of conditional logistic regression models with lasso or elastic net penalties using cross validation

Usage

```
cv.clogitL1 (clObj, numFolds=10)
```

Arguments

`clObj`	an object of type `clogitL1` on which to do cross validation.
`numFolds`	the number of folds used in cross validation. Defaults to the minimum of 10 or the number of observations

Details

Performs `numFolds`-fold cross validation on an object of type `clogitL1`. Using the sequence of regularisation parameters generated by `clObj`, the function chooses *strata* to leave out randomly. The penalised conditional logistic regression model is fit to the non-left-out strata in turn and its deviance compared to an out-of-sample deviance computed on the left-out strata. Fitting models to individual non-left-out strata proceeds using the cyclic coordinate descent-warm start-strong rule type algorithm used in `clogitL1`, only with a prespecified sequence of λ.

Value

An object of type `cv.clogitL1` with the following fields:

`cv_dev`	matrix of size `numLambda`-by-`numFolds` containing the CV deviance in each fold for each value of the regularisation parameter.
`lambda`	vector of regularisation parameters.
`folds`	vector showing the folds membership of each observation.
`mean_cv`	vector containing mean CV deviances for each value of the regularisation parameter.
`se_cv`	vector containing an estimate of the standard error of the CV deviance at each value of the regularisation parameter.
`minCV_lambda`	value of the regularisation parameter at which we have minimum `mean_cv`
`minCV1se_lambda`	value of the regularisation parameter corresponding to the 1-SE rule. Selects the simplest model with estimate CV within 1 standard deviation of the minimum cv.
`nz_beta`	number of nonzero parameter estimates at each value of the regularisation parameter.

References

http://www.jstatsoft.org/v58/i12/

See Also

clogitL1, plot.cv.clogitL1

Examples

```
set.seed(145)

# data parameters
K = 10 # number of strata
n = 5 # number in strata
m = 2 # cases per stratum
p = 20 # predictors

# generate data
y = rep(c(rep(1, m), rep(0, n-m)), K)
X = matrix (rnorm(K*n*p, 0, 1), ncol = p) # pure noise
strata = sort(rep(1:K, n))

par(mfrow = c(1,2))
# fit the conditional logistic model
clObj = clogitL1(y=y, x=X, strata)
plot(clObj, logX=TRUE)

# cross validation
clcvObj = cv.clogitL1(clObj)
plot(clcvObj)
```

---

plot.clogitL1 *Plotting after fitting conditional logistic regression with elastic net penalties*

Description

Takes a clogitL1 object and plots the parameter profile associated with it.

Usage

```
 ## S3 method for class 'clogitL1'
 plot(x, logX=T,
add.legend=F, add.labels=T,
  lty=1:ncol(x$beta), col=1:ncol(x$beta), ...)
```

Arguments

x	an object of type clogitL1.
logX	should the horizontal axis be on log scale?
add.legend	set to TRUE if legend should be printed in top right hand corner. Legend will contain names of variables in data.frame, if specified, otherwise will be numbered from 1 to p in order encountered in original input matrix x
add.labels	set to TRUE if labels are to be added to curves at leftmost side. If variable names are available, these are plotted, otherwise, curves are numbered from 1 to p in order encountered in original input matrix x
lty	usual 'lty' plotting parameter.
col	usual 'col' plotting parameter.
...	additional arguments to plot function

References

http://www.jstatsoft.org/v58/i12/

See Also

clogitL1

Examples

```
set.seed(145)

# data parameters
K = 10 # number of strata
n = 5 # number in strata
m = 2 # cases per stratum
p = 20 # predictors

# generate data
y = rep(c(rep(1, m), rep(0, n-m)), K)
X = matrix (rnorm(K*n*p, 0, 1), ncol = p) # pure noise
strata = sort(rep(1:K, n))

par(mfrow = c(1,2))
# fit the conditional logistic model
clObj = clogitL1(y=y, x=X, strata)
plot(clObj, logX=TRUE)

# cross validation
clcvObj = cv.clogitL1(clObj)
plot(clcvObj)
```

`plot.cv.clogitL1` *Plotting after cross validating conditional logistic regression with elastic net penalties*

Description

Takes a `cv.clogitL1` object and plots the CV deviance curve with standard error bands and minima.

Usage

```
## S3 method for class 'cv.clogitL1'
plot(x, ...)
```

Arguments

`x`	an object of type `cv.clogitL1`.
`...`	additional arguments to `plot` function

References

http://www.jstatsoft.org/v58/i12/

See Also

`cv.clogitL1`

Examples

```
set.seed(145)

# data parameters
K = 10 # number of strata
n = 5 # number in strata
m = 2 # cases per stratum
p = 20 # predictors

# generate data
y = rep(c(rep(1, m), rep(0, n-m)), K)
X = matrix (rnorm(K*n*p, 0, 1), ncol = p) # pure noise
strata = sort(rep(1:K, n))

par(mfrow = c(1,2))
# fit the conditional logistic model
clObj = clogitL1(y=y, x=X, strata)
plot(clObj, logX=TRUE)

# cross validation
clcvObj = cv.clogitL1(clObj)
plot(clcvObj)
```

print.clogitL1 *Printing after fitting conditional logistic regression with elastic net penalties*

Description

Takes a clogitL1 object and prints a summary of the sequence of models fitted.

Usage

```
## S3 method for class 'clogitL1'
print(x, digits = 6, ...)
```

Arguments

x	an object of type clogitL1.
digits	the number of significant digits after the decimal to be printed
...	additional arguments to print function

Details

prints a 3 column data frame with columns:

- Df: number of non-zero parameters in model
- DevPerc: percentage of null deviance explained by current model
- Lambda: associated λ value

References

http://www.jstatsoft.org/v58/i12/

See Also

clogitL1

Examples

```
set.seed(145)

# data parameters
K = 10 # number of strata
n = 5 # number in strata
m = 2 # cases per stratum
p = 20 # predictors

# generate data
y = rep(c(rep(1, m), rep(0, n-m)), K)
X = matrix (rnorm(K*n*p, 0, 1), ncol = p) # pure noise
```

```
strata = sort(rep(1:K, n))

par(mfrow = c(1,2))
# fit the conditional logistic model
clObj = clogitL1(y=y, x=X, strata)
clObj
```

`summary.clogitL1` *Summary after fitting conditional logistic regression with elastic net penalties*

Description

Takes a `clogitL1` object and produces a summary of the sequence of models fitted.

Usage

```
## S3 method for class 'clogitL1'
summary(object, ...)
```

Arguments

`object`	an object of type `clogitL1`.
`...`	any additional arguments passed to `summary` method

Details

Returns a list with a elements `Coefficients`, which holds the matrix of coefficients estimated (each row holding the estimates for a given value of the smoothing parameter) and `Lambda`, which holds the vector of smoothing parameters at which fits were produced.

References

http://www.jstatsoft.org/v58/i12/

See Also

clogitL1

Examples

```
set.seed(145)

# data parameters
K = 10 # number of strata
n = 5 # number in strata
m = 2 # cases per stratum
p = 20 # predictors
```

```
# generate data
y = rep(c(rep(1, m), rep(0, n-m)), K)
X = matrix (rnorm(K*n*p, 0, 1), ncol = p) # pure noise
strata = sort(rep(1:K, n))

par(mfrow = c(1,2))
# fit the conditional logistic model
clObj = clogitL1(y=y, x=X, strata)
summary(clObj)
```

`summary.cv.clogitL1` *Summary after cross validation of conditional logistic regression with elastic net penalties*

Description

Provides summary of conditional logistic regression models after cross validation

Usage

```
## S3 method for class 'cv.clogitL1'
summary(object, ...)
```

Arguments

`object`	an object of type `cv.clogitL1` for which the summary is to be produced.
`...`	additional arguments to `summary` method.

Details

Extracts pertinent information from the supplied `cv.clogitL1` objects. See below for details on output value.

Value

A list with the following fields:

`lambda_minCV`	value of regularisation parameter minimising CV deviance
`beta_minCV`	coefficient profile at the minimising value of the regularisation parameter. Whole dataset used to compute estimates.
`nz_beta_minCV`	number of non-zero coefficients in the CV deviance minimising coefficient profile.
`lambda_minCV1se`	value of regularisaion parameter minimising CV deviance (using 1 standard error rule)
`beta_minCV1se`	coefficient profile at the 1-standard-error-rule value of the regularisation parameter. Whole dataset used to compute estimates.
`nz_beta_minCV1se`	number of non-zero coefficients in the 1-standard-error-rule coefficient profile.

References

http://www.jstatsoft.org/v58/i12/

See Also

clogitL1, plot.cv.clogitL1

Examples

```
set.seed(145)

# data parameters
K = 10 # number of strata
n = 5 # number in strata
m = 2 # cases per stratum
p = 20 # predictors

# generate data
y = rep(c(rep(1, m), rep(0, n-m)), K)
X = matrix (rnorm(K*n*p, 0, 1), ncol = p) # pure noise
strata = sort(rep(1:K, n))

par(mfrow = c(1,2))
# fit the conditional logistic model
clObj = clogitL1(y=y, x=X, strata)
plot(clObj, logX=TRUE)

# cross validation
clcvObj = cv.clogitL1(clObj)
summary(clcvObj)
```

Index