

Package ‘stepPlr’

May 9, 2026

Version 0.93

Date 2018-01-27

Title L2 Penalized Logistic Regression with Stepwise Variable Selection

Author Mee Young Park, Trevor Hastie

Maintainer Mee Young Park <meeyoung@google.com>

Depends R (>= 2.0)

Description L2 penalized logistic regression for both continuous and discrete predictors, with forward stagewise/forward stepwise variable selection procedure.

License GPL (>= 2)

Repository CRAN

Date/Publication 2018-01-28 16:04:11 UTC

NeedsCompilation yes

Contents

cv.step.plr	1
plr	3
predict.plr	5
predict.stepplr	6
step.plr	7

Index	10
--------------	-----------

cv.step.plr	<i>Computes cross-validated deviance or prediction errors for step.plr</i>
-------------	--

Description

This function computes cross-validated deviance or prediction errors for `step.plr`. The parameters that can be cross-validated are `lambda` and `cp`.

Usage

```
cv.step.plr(x, y, weights = rep(1, length(y)),  
            nfold = 5, folds = NULL, lambda = c(1e-4, 1e-2, 1),  
            cp = c("aic", "bic"), cv.type=c("deviance", "class"),  
            trace = TRUE, ...)
```

Arguments

x	matrix of features
y	binary response
weights	optional vector of weights for observations
nfold	number of folds to be used in cross-validation. Default is nfold=5.
folds	list of cross-validation folds. Its length must be nfold. If NULL, the folds are randomly generated.
lambda	vector of the candidate values for lambda in step.plr
cp	vector of the candidate values for cp in step.plr
cv.type	If cv.type=deviance, cross-validated deviances are returned. If cv.type=class, cross-validated prediction errors are returned.
trace	If TRUE, the steps are printed out.
...	other options for step.plr

Details

This function computes cross-validated deviance or prediction errors for step.plr. The parameters that can be cross-validated are lambda and cp. If both are input as vectors (of length greater than 1), then a two-dimensional cross-validation is done. If either one is input as a single value, then the cross-validation is done only on the parameter with multiple inputs.

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2008) Penalized Logistic Regression for Detecting Gene Interactions

See Also

step.plr

Examples

```
n <- 100
p <- 5
x <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
y <- sample(c(0, 1), n, replace=TRUE)
level <- vector("list", length=p)
for (i in 1:p) level[[i]] <- seq(3)
cvfit <- cv.step.plr(x, y, level=level, lambda=c(1e-4, 1e-2, 1), cp="bic")
```

plr

*Logistic regression with a quadratic penalization on the coefficients***Description**

This function fits a logistic regression model penalizing the size of the L2 norm of the coefficients.

Usage

```
plr(x, y, weights = rep(1,length(y)),
    offset.subset = NULL, offset.coefficients = NULL,
    lambda = 1e-4, cp = "bic")
```

Arguments

x	matrix of features
y	binary response
weights	optional vector of weights for observations
offset.subset	optional vector of indices for the predictors for which the coefficients are preset to offset.coefficients. If offset.coefficients is not NULL, offset.subset must be provided.
offset.coefficients	optional vector of preset coefficient values for the predictors in offset.subset. If offset.coefficient is not NULL, offset.coefficients must be provided.
lambda	regularization parameter for the L2 norm of the coefficients. The minimizing criterion in plr is $-\log\text{-likelihood} + \lambda * \ \beta\ ^2$. Default is $\lambda=1e-4$.
cp	complexity parameter to be used when computing the score. $\text{score} = \text{deviance} + \text{cp} * \text{df}$. If cp="aic" or cp="bic", these are converted to cp=2 or cp=log(sample size), respectively. Default is cp="bic".

Details

We proposed using logistic regression with a quadratic penalization on the coefficients for detecting gene interactions as described in "Penalized Logistic Regression for Detecting Gene Interactions (2008)" by Park and Hastie. However, this function plr may be used for a general purpose.

Value

A plr object is returned. `predict`, `print`, and `summary` functions can be applied.

<code>coefficients</code>	vector of the coefficient estimates
<code>covariance</code>	sandwich estimate of the covariance matrix for the coefficients
<code>deviance</code>	deviance of the fitted model
<code>null.deviance</code>	deviance of the null model
<code>df</code>	degrees of freedom of the fitted model
<code>score</code>	deviance + $cp \cdot df$
<code>nobs</code>	number of observations
<code>cp</code>	complexity parameter used when computing the score
<code>fitted.values</code>	fitted probabilities
<code>linear.predictors</code>	linear predictors computed with the estimated coefficients
<code>level</code>	If any categorical factors are input, <code>level</code> - the list of level sets - is automatically generated and returned. See <code>step.plr</code> for details of how it is generated.

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2008) Penalized Logistic Regression for Detecting Gene Interactions

See Also

`predict.plr`, `step.plr`

Examples

```
n <- 100

p <- 10
x <- matrix(rnorm(n * p), nrow=n)
y <- sample(c(0, 1), n, replace=TRUE)
fit <- plr(x, y, lambda=1)

p <- 3
z <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
x <- data.frame(x1=factor(z[, 1]), x2=factor(z[, 2]), x3=factor(z[, 3]))
y <- sample(c(0, 1), n, replace=TRUE)
fit <- plr(x, y, lambda=1)
# 'level' is automatically generated. Check 'fit$level'.
```

predict.plr *prediction function for plr*

Description

This function computes the linear predictors, probability estimates, or the class labels for new data, using a `plr` object.

Usage

```
## S3 method for class 'plr'
predict(object, newx = NULL,
        type = c("link", "response", "class"), ...)
```

Arguments

<code>object</code>	<code>plr</code> object
<code>newx</code>	matrix of features at which the predictions are made. If <code>newx=NULL</code> , predictions for the training data are returned.
<code>type</code>	If <code>type=link</code> , the linear predictors are returned; if <code>type=response</code> , the probability estimates are returned; and if <code>type=class</code> , the class labels are returned. Default is <code>type=link</code> .
<code>...</code>	other options for prediction

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2008) Penalized Logistic Regression for Detecting Gene Interactions

See Also

`plr`

Examples

```
n <- 100

p <- 10
x0 <- matrix(rnorm(n * p), nrow=n)
y <- sample(c(0, 1), n, replace=TRUE)
fit <- plr(x0, y, lambda=1)
x1 <- matrix(rnorm(n * p), nrow=n)
pred1 <- predict(fit, x1, type="link")
pred2 <- predict(fit, x1, type="response")
```

```

pred3 <- predict(fit, x1, type="class")

p <- 3
z <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
x0 <- data.frame(x1=factor(z[, 1]), x2=factor(z[, 2]), x3=factor(z[, 3]))
y <- sample(c(0, 1), n, replace=TRUE)
fit <- plr(x0, y, lambda=1)
z <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
x1 <- data.frame(x1=factor(z[, 1]), x2=factor(z[, 2]), x3=factor(z[, 3]))
pred1 <- predict(fit, x1, type="link")
pred2 <- predict(fit, x1, type="response")
pred3 <- predict(fit, x1, type="class")

```

predict.stepplr *prediction function for step.plr*

Description

This function computes the linear predictors, probability estimates, or the class labels for new data, using a stepplr object.

Usage

```

## S3 method for class 'stepplr'
predict(object, x = NULL, newx = NULL,
        type = c("link", "response", "class"), ...)

```

Arguments

object	stepplr object
x	matrix of features used for fitting object. If newx is provided, x must be provided as well.
newx	matrix of features at which the predictions are made. If newx=NULL, predictions for the training data are returned.
type	If type=link, the linear predictors are returned; if type=response, the probability estimates are returned; and if type=class, the class labels are returned. Default is type=link.
...	other options for prediction

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2008) Penalized Logistic Regression for Detecting Gene Interactions

See Also

stepplr

Examples

```

n <- 100
p <- 5
x0 <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
x0 <- cbind(rnorm(n), x0)
y <- sample(c(0, 1), n, replace=TRUE)
level <- vector("list", length=6)
for (i in 2:6) level[[i]] <- seq(3)
fit <- step.plr(x0, y, level=level)
x1 <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
x1 <- cbind(rnorm(n), x1)
pred1 <- predict(fit, x0, x1, type="link")
pred2 <- predict(fit, x0, x1, type="response")
pred3 <- predict(fit, x0, x1, type="class")

```

step.plr

*Forward stepwise selection procedure for penalized logistic regression***Description**

This function fits a series of L2 penalized logistic regression models selecting variables through the forward stepwise selection procedure.

Usage

```

step.plr(x, y, weights = rep(1,length(y)), fix.subset = NULL,
         level = NULL, lambda = 1e-4, cp = "bic", max.terms = 5,
         type = c("both", "forward", "forward.stagewise"),
         trace = FALSE)

```

Arguments

<code>x</code>	matrix of features
<code>y</code>	binary response
<code>weights</code>	optional vector of weights for observations
<code>fix.subset</code>	vector of indices for the variables that are forced to be in the model
<code>level</code>	list of length <code>ncol(x)</code> . The j -th element corresponds to the j -th column of <code>x</code> . If the j -th column of <code>x</code> is discrete, <code>level[[j]]</code> is the set of levels for the categorical factor. If the j -th column of <code>x</code> is continuous, <code>level[[j]] = NULL</code> . <code>level</code> is automatically generated in the function; however, if any levels of the categorical factors are not observed, but still need to be included in the model, then the user must provide the complete sets of the levels through <code>level</code> . If a numeric column needs to be considered discrete, it can be done by manually providing <code>level</code> as well.

lambda	regularization parameter for the L2 norm of the coefficients. The minimizing criterion in plr is $-\log\text{-likelihood} + \lambda * \ \beta\ ^2$. Default is lambda=1e-4.
cp	complexity parameter to be used when computing the score. score=deviance+cp*df. If cp="aic" or cp="bic", these are converted to cp=2 or cp=log(sample size), respectively. Default is cp="bic".
max.terms	maximum number of terms to be added in the forward selection procedure. Default is max.terms=5.
type	If type="both", forward selection is followed by a backward deletion. If type="forward", only a forward selection is done. If type="forward.stagewise", variables are added in the forward-stagewise method. Default is "both".
trace	If TRUE, the variable selection procedure prints out its progress.

Details

This function implements an L2 penalized logistic regression along with the stepwise variable selection procedure, as described in "Penalized Logistic Regression for Detecting Gene Interactions (2008)" by Park and Hastie.

If type="forward", max.terms terms are sequentially added to the model, and the model that minimizes score is selected as the optimal fit. If type="both", a backward deletion is done in addition, which provides a series of models with a different combination of the selected terms. The optimal model minimizing score is chosen from the second list.

Value

A `stepplr` object is returned. `anova`, `predict`, `print`, and `summary` functions can be applied.

fit	plr object for the optimal model selected
action	list that stores the selection order of the terms in the optimal model
action.name	list of the names of the sequentially added terms - in the same order as in action
deviance	deviance of the fitted model
df	residual degrees of freedom of the fitted model
score	deviance + cp*df, where df is the model degrees of freedom
group	vector of the counts for the dummy variables, to be used in <code>predict.stepplr</code>
y	response variable used
weight	weights used
fix.subset	fix.subset used
level	level used
lambda	lambda used
cp	complexity parameter used when computing the score
type	type used
xnames	column names of x

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2008) Penalized Logistic Regression for Detecting Gene Interactions

See Also

cv.step.plr, plr, predict.stepplr

Examples

```
n <- 100

p <- 3
z <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
x <- data.frame(x1=factor(z[, 1]), x2=factor(z[, 2]), x3=factor(z[, 3]))
y <- sample(c(0, 1), n, replace=TRUE)
fit <- step.plr(x, y)
# 'level' is automatically generated. Check 'fit$level'.

p <- 5
x <- matrix(sample(seq(3), n * p, replace=TRUE), nrow=n)
x <- cbind(rnorm(n), x)
y <- sample(c(0, 1), n, replace=TRUE)
level <- vector("list", length=6)
for (i in 2:6) level[[i]] <- seq(3)
fit1 <- step.plr(x, y, level=level, cp="aic")
fit2 <- step.plr(x, y, level=level, cp=4)
fit3 <- step.plr(x, y, level=level, type="forward")
fit4 <- step.plr(x, y, level=level, max.terms=10)
# This is an example in which 'level' was input manually.
# level[[1]] should be either 'NULL' or 'NA' since the first factor is continuous.
```

Index

* **models**

- cv.step.plr, 1
- plr, 3
- predict.plr, 5
- predict.stepplr, 6
- step.plr, 7

* **regression**

- cv.step.plr, 1
- plr, 3
- predict.plr, 5
- predict.stepplr, 6
- step.plr, 7

cv.step.plr, 1

plr, 3
predict.plr, 5
predict.stepplr, 6

step.plr, 7