Package 'penalizedclr'

April 22, 2021

Title Integrative Penalized Conditional Logistic Regression

Version 0.1.0

Description Implements L1 and L2 penalized conditional logistic regression with penalty factors allowing for integration of multiple data sources. Implements stability selection for variable selection.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

Imports penalized, survival, clogitL1, stats, tidyverse

Suggests parallel, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Vera Djordjilovi'c [aut, cre] (<https://orcid.org/0000-0002-7670-3111>), Erica Ponzi [aut]

Maintainer Vera Djordjilovi'c <vera.djordjilovic@unive.it>

Repository CRAN

Date/Publication 2021-04-22 07:30:02 UTC

R topics documented:

`default.lambda` *Default values for L1 penalty in conditional logistic regression*

Description

Performs cross validation to determine reasonable default values for L1 penalty in a conditional logistic regression

Usage

```
default.lambda(X, Y, stratum, nfolds = 10, alpha = 1)
```

Arguments

`X`	A matrix of covariates, with the number of rows equalling the number of observations.
`Y`	A binary response variable.
`stratum`	A numeric vector with stratum membership of each observation.
`nfolds`	The number of folds used in cross-validation. Default is 10.
`alpha`	The elastic net mixing parameter, a number between 0 and 1. alpha=0 would give pure ridge; alpha=1 gives lasso. Pure ridge penalty is never obtained in this implementation since alpha must be positive.

Value

A numeric vector of length 1 to 3 (depending on the problem) giving L1 penalties

`find.default.lambda` *Default values for L1 penalty in conditional logistic regression*

Description

Performs cross validation to determine reasonable default values for L1 penalty in a conditional logistic regression

Usage

```
find.default.lambda(
  response,
  stratum,
  penalized,
  unpenalized = NULL,
  alpha = 1,
  p = NULL,
```

```
  standardize = FALSE,
  event,
  nfolds = 10
)
```

Arguments

response	The response variable, either a 0/1 vector or a factor with two levels.
stratum	A numeric vector with stratum membership of each observation.
penalized	A matrix of penalized covariates.
unpenalized	A matrix of additional unpenalized covariates.
alpha	The elastic net mixing parameter, a number between 0 and 1. alpha=0 would give pure ridge; alpha=1 gives lasso. Pure ridge penalty is never obtained in this implementation since alpha must be positive.
p	The sizes of blocks of covariates, a numerical vector of the length equal to the number of blocks, and with the sum equal to the number of penalized covariates. If missing, all covariates are treated the same and a single penalty is applied.
standardize	Should the covariates be standardized, a logical value.
event	If response is a factor, the level that should be considered a success in the logistic regression.
nfolds	The number of folds used in cross-validation. Default is 10.

Details

The function `find.default.lambda` is used to obtain a sequence of reasonable values of the parameter `lambda`. In the presence of blocks of covariates, a separate sequence is obtained for each block and the function returns a list. The function is based on cross-validation implemented in the `clogitL1` package. For each block, a separate conditional logistic model is fit (together with unpenalized covariates if provided) and two `lambda` values a) `minCV_lambda` minimizing cross-validated deviance and b) `minCV1se_lambda` satisfying 1-SE rule, are obtained. Two additional values of `lambda` are obtained as a midpoint between the two (`minCV_lambda` + `minCV1se_lambda`)`/2` and symmetrically (3*minCV_lambda - minCV1se_lambda)/2) (on a log-scale). If `minCV_lambda` = `minCV1se_lambda`, a shorter sequnce is returned. Note that cross-validation includes random data splitting, meaning that obtained values can vary significantly between different runs.

Value

A numeric vector giving a sequence of L1 penalties if p is missing, or a list of per-block penalty sequences otherwise.

Examples

```
set.seed(123)
# simulate covariates (pure noise in two blocks of 20 and 80 variables)
X <- cbind(matrix(rnorm(4000, 0, 1), ncol = 20), matrix(rnorm(16000, 2, 0.6), ncol = 80))
p <- c(20,80)
# stratum membership
```

```
stratum <- sort(rep(1:100, 2))

# the response
Y <- rep(c(1, 0), 100)

# obtain a list with a separate sequence for each block

lambda.list <- find.default.lambda(response = Y,
                                   penalized = X, stratum = stratum, p = p)

# obtain a single sequence

lambda.seq <- find.default.lambda(response = Y,
                                   penalized = X, stratum = stratum)
```

`penalized.clr` *Penalized conditional logistic regression*

Description

Fits conditional logistic regression models with L1 and L2 penalty allowing for different penalties for different blocks of covariates.

Usage

```
penalized.clr(
  response,
  stratum,
  penalized,
  unpenalized = NULL,
  lambda = NULL,
  alpha = 1,
  p = NULL,
  standardize = FALSE,
  event
)
```

Arguments

`response`	The response variable, either a 0/1 vector or a factor with two levels.
`stratum`	A numeric vector with stratum membership of each observation.
`penalized`	A matrix of penalized covariates.
`unpenalized`	A matrix of additional unpenalized covariates.
`lambda`	The tuning parameter for L1. Either a single non-negative number, or a numeric vector of the length equal to the number of blocks. If NULL, function `find.default.lambda` is called. See p below.

alpha	The elastic net mixing parameter, a number between 0 and 1. alpha=0 would give pure ridge; alpha=1 gives lasso. Pure ridge penalty is never obtained in this implementation since alpha must be positive.
p	The sizes of blocks of covariates, a numerical vector of the length equal to the number of blocks, and with the sum equal to the number of penalized covariates. If missing, all covariates are treated the same and a single penalty is applied.
standardize	Should the covariates be standardized, a logical value.
event	If response is a factor, the level that should be considered a success in the logistic regression.

Details

The `penalized.clr` function fits a conditional logistic regression model for a given combination of L1 (`lambda`) and L2 penalties. L2 penalty is obtained from `lambda` and `alpha` as `lambda*(1-alpha)/(2*alpha)`. Note that `lambda` is a single number if all covariates are to be penalized equally, and a vector of penatlies, if predictors are divided in blocks (of sizes provided in `p`) that are to be penalized differently. If `lambda` is not provided by the user, a default value is computed by the `find.default.lambda` function (which slows down the computation). The `penalized.clr` function is based on the Cox model routine available in the `penalized` package.

Value

A list with the following elements:

- `penalized` - Regression coefficients for the penalized covariates.
- `unpenalized` - Regression coefficients for the unpenalized covariates.
- `converged` - Whether the fitting process was judged to be converged.
- `lambda` - The tuning parameter for L1 used.
- `alpha` - The elastic net mixing parameter used.

See Also

`stable.clr` and `stable.clr.g` for variable selection through stability selection in penalized conditional logistic regression with a single penalty factor and multiple penalty factors, respectively.

Examples

```
set.seed(123)
# simulate covariates (pure noise in two blocks of 20 and 80 variables)
X <- cbind(matrix(rnorm(4000, 0, 1), ncol = 20), matrix(rnorm(16000, 2, 0.6), ncol = 80))

# stratum membership
stratum <- sort(rep(1:100, 2))

# the response
Y <- rep(c(1, 0), 100)

fit <- penalized.clr( response = Y, stratum = stratum,
  penalized = X, lambda = c(1, 0.3),
```

```
  p = c(20, 80), standardize = TRUE)
fit$penalized
fit$converged
fit$lambda
```

`stable.clr` *Stability selection based on penalized conditional logistic regression*

Description

Performs stability selection for conditional logistic regression models with L1 and L2 penalty.

Usage

```
stable.clr(
  response,
  stratum,
  penalized,
  unpenalized = NULL,
  lambda.seq = NULL,
  alpha = 1,
  B = 100,
  parallel = TRUE,
  standardize = FALSE,
  event
)
```

Arguments

`response`	The response variable, either a 0/1 vector or a factor with two levels.
`stratum`	A numeric vector with stratum membership of each observation.
`penalized`	A matrix of penalized covariates.
`unpenalized`	A matrix of additional unpenalized covariates.
`lambda.seq`	a sequence of non-negative values to be used as tuning parameters for L1
`alpha`	The elastic net mixing parameter, a number between 0 and 1. alpha=0 would give pure ridge; alpha=1 gives lasso. Pure ridge penalty is never obtained in this implementation since alpha must be positive.
`B`	A single positive number for the number of subsamples.
`parallel`	Logical. Should the computation be parallelized?
`standardize`	Should the covariates be standardized, a logical value.
`event`	If response is a factor, the level that should be considered a success in the logistic regression.

Value

A list with a numeric vector `Pilambda` giving selection probabilities for each penalized covariate, and a sequence `lambda.seq` used.

See Also

`stable.clr.g` for stability selection in penalized conditional logistic regression with multiple penalties for block structured covariates.

Examples

```
 set.seed(123)

# simulate covariates (pure noise in two blocks of 20 and 80 variables)
X <- cbind(matrix(rnorm(4000, 0, 1), ncol = 20), matrix(rnorm(16000, 2, 0.6), ncol = 80))

# stratum membership
stratum <- sort(rep(1:100, 2))

# the response
Y <- rep(c(1, 0), 100)

# sequence of L1 penalties
lambda.seq <- find.default.lambda(response = Y,
                                  penalized = X,
                                  stratum = stratum)

# perform stability selection

stable1 <- stable.clr(response = Y, penalized = X, stratum = stratum,
                          lambda.seq = lambda.seq)

# when lambda.seq is not provided,
# it is computed within the function (slightly different results might occur due to the
# randomness inherent to cross-validation)


stable2 <- stable.clr.g(response = Y, penalized = X, stratum = stratum)
```

`stable.clr.g` *Stability selection based on penalized conditional logistic regression*

Description

Performs stability selection for conditional logistic regression models with L1 and L2 penalty allowing for different penalties for different blocks (groups) of covariates (different data sources).

Usage

```
stable.clr.g(
  response,
  stratum,
  penalized,
  unpenalized = NULL,
  p = NULL,
  lambda.list = NULL,
  alpha = 1,
  B = 100,
  parallel = TRUE,
  standardize = FALSE,
  event
)
```

Arguments

response	The response variable, either a 0/1 vector or a factor with two levels.
stratum	A numeric vector with stratum membership of each observation.
penalized	A matrix of penalized covariates.
unpenalized	A matrix of additional unpenalized covariates.
p	The sizes of blocks of covariates, a numerical vector of the length equal to the number of blocks, and with the sum equal to the number of penalized covariates. If missing, all covariates are treated the same and a single penalty is applied.
lambda.list	A list with per-block sequences of L1 penalties. If NULL, `find.default.lambda` function is called.
alpha	The elastic net mixing parameter, a number between 0 and 1. alpha=0 would give pure ridge; alpha=1 gives lasso. Pure ridge penalty is never obtained in this implementation since alpha must be positive.
B	A single positive number for the number of subsamples.
parallel	Logical. Should the computation be parallelized?
standardize	Should the covariates be standardized, a logical value.
event	If response is a factor, the level that should be considered a success in the logistic regression.

Details

This function implements stability selection (Meinshausen and Bühlmann, 2010) in a conditional logistic regression. The implementation is based on the modification of Shah and Samworth (2013) featuring complementary subsamples. Note that this means that the number of subsamples will be 2B instead of B. Subsampling procedure is repeated 2B times for each combination of per-block penalties resulting each time in a vector of selection frequences (frequency of non-zero coefficient estimate of each covariate). The final selection probability `Pilambda` is obtained by taking the maximum over all considered values of penalties.

Value

A list containing a numeric vector `Pilambda`, giving selection probabilities for all penalized covariates and `lambda.list`.

References

1. Meinshausen, N., & Bühlmann, P. (2010). Stability selection. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(4), 417-473.
2. Shah, R. D., & Samworth, R. J. (2013). Variable selection with error control: another look at stability selection. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 75(1), 55-80.

See Also

`find.default.lambda` for obtaining default sequences of L1 penalties.

Examples

```
set.seed(123)

# simulate covariates (pure noise in two blocks of 20 and 80 variables)
X <- cbind(matrix(rnorm(4000, 0, 1), ncol = 20), matrix(rnorm(16000, 2, 0.6), ncol = 80))
p <- c(20,80)

# stratum membership
stratum <- sort(rep(1:100, 2))

# the response
Y <- rep(c(1, 0), 100)

# list of L1 penalties
lambda.list <- find.default.lambda(response = Y,
                                   penalized = X, stratum = stratum, p = p)

# perform stability selection

stable.g1 <- stable.clr.g(response = Y, penalized = X, stratum = stratum,
                          p = p, lambda.list = lambda.list)

# when lambda.list is not provided,
# it is computed within the function (slightly different results might occur due to the
# randomness inherent to cross-validation)

stable.g2 <- stable.clr.g(response = Y, penalized = X, stratum = stratum,
                           p = p)

# if p is not provided, all covariates are penalized equally

stable.g3 <- stable.clr.g(response = Y, penalized = X, stratum = stratum)
```

`subsample.clr` *Stability selection for penalized conditional logistic regression*

Description

Stability selection for penalized conditional logistic regression

Usage

```
subsample.clr(
  response,
  stratum,
  penalized,
  unpenalized = NULL,
  lambda,
  alpha = 1,
  B = 100,
  matB = NULL,
  return.matB = FALSE,
  parallel = TRUE,
  standardize = FALSE
)
```

Arguments

`response`	The response variable, either a 0/1 vector or a factor with two levels.
`stratum`	A numeric vector with stratum membership of each observation.
`penalized`	A matrix of penalized covariates.
`unpenalized`	A matrix of additional unpenalized covariates.
`lambda`	The tuning parameter for L1. Either a single non-negative number, or a numeric vector of the length equal to the number of blocks. If NULL, function `find.default.lambda` is called. See p below.
`alpha`	The elastic net mixing parameter, a number between 0 and 1. alpha=0 would give pure ridge; alpha=1 gives lasso. Pure ridge penalty is never obtained in this implementation since alpha must be positive.
`B`	A single positive number for the number of subsamples.
`matB`	A 2B x ceiling(unique(stratum)/2) matrix with index set of selected strata in each of 2B subsamples
`return.matB`	Logical. Should the matrix matB be returned?
`parallel`	Logical. Should the computation be parallelized?
`standardize`	Should the covariates be standardized, a logical value.

Value

If `return.matB` is TRUE, a list with two elements, a numeric vector Pilambda, giving selection probabilities for each covariate and a matrix `matB`; otheriwise only Pilambda.

Index