Package 'ordinalNet'

July 27, 2016

Type Package

Title Penalized Ordinal Regression

Version 1.5

Date 2016-07-27

Description Fits ordinal regression models with elastic net penalty. Supported models include cumulative logit, probit, cauchit, and complementary log-log. The algorithm uses Fisher Scoring with Coordinate Descent updates.

LazyData TRUE

License MIT + file LICENSE

Imports stats

Suggests testthat

RoxygenNote 5.0.1

NeedsCompilation no

Author Michael Wurm [aut, cre],
Bret Hanlon [ths]

Maintainer Michael Wurm <wurm@wisc.edu>

Repository CRAN

Date/Publication 2016-07-27 10:58:34

R topics documented:

`coef.ordinalNetFit` *Extracts fitted coefficients from an "ordinalNetFit" object.*

Description

Extracts fitted coefficients from an "ordinalNetFit" object.

Usage

```
## S3 method for class 'ordinalNetFit'
coef(object, whichLambda = NULL, criteria = c("aic",
  "bic"), ...)
```

Arguments

`object`	An "ordinalNetFit" S3 object.
`whichLambda`	Optional index number(s) of the desired `lambda` within the sequence of `lambdaVals`. `whichLambda=1:nLambda` will return the entire solution path.
`criteria`	Selects the best `lambda` value by AIC or BIC. Only used if `whichLambda=NULL`.
`...`	Not used. Additional coef arguments.

Value

Vector of fitted coefficients.

Examples

```
set.seed(10)
x <- matrix(rnorm(50*5), ncol=5)
beta <- c(1, 0, 0, 0, 0)
intercepts <- c(-1, 1)
xb <- x %*% beta
eta <- cbind(xb+intercepts[1], xb+intercepts[2])
probMatCumul <- 1 / (1 + exp(-eta))
probMat <- cbind(probMatCumul, 1) - cbind(0, probMatCumul)
y <- apply(probMat, 1, function(p) sample(1:3, size=1, prob=p))
y <- as.factor(y)
fit <- ordinalNet(x, y)
coef(fit)
```

`ordinalNet` *Penalized ordinal regression*

Description

Fits ordinal regression models with elastic net penalty. Supported models include cumulative logit, probit, cauchit, and complementary log-log. The regularization path is computed at a grid of values for the regularizaton parameter `lambda`. The algorithm uses Fisher Scoring with Coordinate Descent updates.

Usage

```
ordinalNet(x, y, alpha = 1, standardize = TRUE, penalizeID = NULL,
  positiveID = NULL, link = c("logit", "probit", "cloglog", "cauchit"),
  lambdaVals = NULL, nLambda = 100, lambdaMinRatio = ifelse(nObs < nVar,
  0.01, 1e-04), alphaMin = 0.01, trace = FALSE, epsOut = 0.001,
  epsIn = 0.001, maxiterOut = Inf, maxiterIn = Inf, pMin = 1e-20,
  betaMin = 1e-08, convNorm = Inf, zetaStart = NULL, thetaStart = NULL)
```

Arguments

`x`	Covariate matrix.
`y`	Response variable. Either an ordered factor or a matrix where each row is a multinomial vector of counts.
`alpha`	The elastic net mixing parameter, with $0 \leq \alpha \leq 1$. `alpha=1` is the lasso penalty, and `alpha=0` is the ridge penalty. See "Details".
`standardize`	If `standardize=TRUE`, the predictor variables are scaled to have unit variance. Coefficient estimates are returned on the original scale.
`penalizeID`	Logical vector indicating whether each coefficient should be penalized. Default is TRUE for all coefficients.
`positiveID`	Logical vector indicating whether each coefficient should be constrained to be non-negative. Default is FALSE for all coefficients.
`link`	Specifies the link function. The options supported are logit, probit, complementary log-log, and cauchit.
`lambdaVals`	An optional user-specified lambda sequence. Typical usage is to have the program compute its own `lambda` sequence based on `nLambda` and `lambdaMinRatio`.
`nLambda`	The number of `lambda` values in the solution path. Default is 100.
`lambdaMinRatio`	Smallest value for `lambda` as a fraction of the maximum lambda. (The maximum lambda is the smallest value that sets all penalized coefficients to zero.)
`alphaMin`	If `alpha < alphaMin`, then `alphaMin` is used to calculate the maximum lambda value.
`trace`	If `trace=TRUE` the algorithm progress is printed to the terminal.

epsOut	Convergence threshold for the algorithm's outer loop. The outer loop optimizes optimizes the Fisher Scoring quadratic approximation to the penalized log likelihood. The default value is relatively high to speed up computation time. It is recommended to keep epsOut equal to epsIn.
epsIn	Convergence threshold for the algorithm's inner loop. The inner loop cycles through and updates the coefficient estimates using coordinate descent. Each cycle is one iteration. The default value is relatively high to speed up computation time. It is recommended to keep epsOut equal to epsIn.
maxiterOut	Maximum number of outer loop iterations.
maxiterIn	Maximum number of inner loop iterations.
pMin	If for any observation, the fitted probability for a response category falls below pMin, the algorithm is terminated. This can occur for small lambda values as the coefficient estimates diverge to $+/-\infty$.
betaMin	If a coefficient estimate falls below betaMin, it is set to zero. This improves the stability and speed of the fitting algorithm.
convNorm	The L-p norm of the coefficient estimate relative changes is computed after each iteration of the inner or outer loop. Convergence of the loop is assessed by comparing this value to the convergence threshold (epsIn or epsOut). convNorm=Inf is the default and represents the L-∞ norm (maximum relative change).
zetaStart	Optional user-specified starting values for the intercept terms (must be non-decreasing). Default is a uniform sequency from -1 to 1.
thetaStart	Optional user-specified starting values for the non-intercept terms. Default is zero for all coefficients.

Details

The ordinal model has the form

$$g(P(y \leq j|x)) = Intercept[j] + x\beta,$$

where $g(.)$ is a link function, most commonly logit.

The elastic net penalty is defined as

$$\lambda(1-\alpha)/2||\beta||_2^2 + \alpha||\beta||_1.$$

The objective function is

$$-1/N * loglik + penalty.$$

Value

An object with S3 class "ordinalNetFit". This includes a matrix of coefficient estimates (if covariates were scaled with standardize=TRUE, the coefficients are returned on the original scale). It also includes a list called "mirlsNetFit" containing arguments used in the multiple iteratively re-weighted least squares optimization routine.

Examples

```
set.seed(10)
x <- matrix(rnorm(50*5), ncol=5)
beta <- c(1, 0, 0, 0, 0)
intercepts <- c(-1, 1)
xb <- x %*% beta
eta <- cbind(xb+intercepts[1], xb+intercepts[2])
probMatCumul <- 1 / (1 + exp(-eta))
probMat <- cbind(probMatCumul, 1) - cbind(0, probMatCumul)
y <- apply(probMat, 1, function(p) sample(1:3, size=1, prob=p))
y <- as.factor(y)
fit <- ordinalNet(x, y)
print(fit)
coef(fit)
predict(fit, type="class")
predict(fit, type="prob")
```

---

predict.ordinalNetFit *Predict method for an "ordinalNetFit" object*

Description

Obtains predicted class numbers or fitted class probabilities for either the model matrix used to fit the model or a new model matrix.

Usage

```
## S3 method for class 'ordinalNetFit'
predict(object, newx = NULL, whichLambda = NULL,
  criteria = c("aic", "bic"), type = c("class", "prob"), ...)
```

Arguments

`object`	An "ordinalNetFit" S3 object.
`newx`	Optional covariate matrix. If the model was fit with a centered and scaled matrix, then `newx` should be centered and scaled as well.
`whichLambda`	Optional index number of the desired `lambda` within the sequence of `lambda` values in the solution path.
`criteria`	Selects the best `lambda` value by AIC or BIC. Only used if `whichLambda=NULL`.
`type`	Specifies whether to return a vector of predicted class numbers or a matrix of fitted class probabilities.
`...`	Not used. Additional predict arguments.

Value

A vector of predicted class numbers or a matrix of fitted class probabilities, depending on `type`.

Examples

```
set.seed(10)
x <- matrix(rnorm(50*5), ncol=5)
beta <- c(1, 0, 0, 0, 0)
intercepts <- c(-1, 1)
xb <- x %*% beta
eta <- cbind(xb+intercepts[1], xb+intercepts[2])
probMatCumul <- 1 / (1 + exp(-eta))
probMat <- cbind(probMatCumul, 1) - cbind(0, probMatCumul)
y <- apply(probMat, 1, function(p) sample(1:3, size=1, prob=p))
y <- as.factor(y)
fit <- ordinalNet(x, y)
predict(fit, type="class")
predict(fit, type="prob")
```

`print.ordinalNetFit` *Print method for an "ordinalNetFit" object.*

Description

Provides a model fit summary in matrix form. For each `lambda` value in the solution path, the following information is included: degrees of freedom (number of nonzero parameters), AIC, BIC, and percent deviance explained.

Usage

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

Arguments

x	An "ordinalNetFit" S3 object.
...	Not used. Additional print arguments.

Value

Silently returns the model fit summary.

Examples

```
set.seed(10)
x <- matrix(rnorm(50*5), ncol=5)
beta <- c(1, 0, 0, 0, 0)
intercepts <- c(-1, 1)
xb <- x %*% beta
eta <- cbind(xb+intercepts[1], xb+intercepts[2])
probMatCumul <- 1 / (1 + exp(-eta))
probMat <- cbind(probMatCumul, 1) - cbind(0, probMatCumul)
y <- apply(probMat, 1, function(p) sample(1:3, size=1, prob=p))
```

```
y <- as.factor(y)
fit <- ordinalNet(x, y)
print(fit)
```

Index