

Package ‘glmpath’

April 17, 2009

Version 0.94

Date 2007-10-07

Title L1 Regularization Path for Generalized Linear Models and Cox Proportional Hazards Model

Author Mee Young Park, Trevor Hastie

Maintainer Mee Young Park <meeyoung@google.com>

Depends survival, R (>= 2.0)

Description A path-following algorithm for L1 regularized generalized linear models and Cox proportional hazards model

License GPL-2

OS_type unix

Repository CRAN

Date/Publication 2007-10-08 08:55:48

R topics documented:

bootstrap.path	2
coxpath	3
cv.coxpath	6
cv.glmpath	7
glmpath	8
heart.data	11
lung.data	12
plot.bootpath	13
plot.coxpath	14
plot.glmpath	16
predict.coxpath	17
predict.glmpath	19
summary.coxpath	20
summary.glmpath	21

Index	23
--------------	-----------

bootstrap.path	<i>Generates a set of bootstrap coefficients for either glm path or cox path</i>
----------------	--

Description

This function generates a set of bootstrap coefficients for either `glm path` or `cox path`. For each bootstrap run, the regularization parameter may be determined based on either `aic` or `bic`.

Usage

```
bootstrap.path(x, y, data, B, index=NULL, path=c("glm path", "cox path"),
              method=c("aic", "bic"), trace=FALSE, ...)
```

Arguments

<code>x</code>	matrix of features
<code>y</code>	response
<code>data</code>	a list of data components. If <code>path=glm path</code> , <code>data</code> consists of <code>x</code> : a matrix of features and <code>y</code> : response. <code>data</code> is not needed if <code>x</code> and <code>y</code> are input separately. If <code>path=cox path</code> , <code>data</code> must be provided, including <code>x</code> : a matrix of features, <code>time</code> : the survival time, and <code>status</code> : censor status with 1 if died and 0 if censored.
<code>B</code>	number of bootstrap runs
<code>index</code>	matrix (<code>B</code> rows, <code>ncol(x)</code> columns) of bootstrap sample indices. Each row is a vector of indices for a bootstrap run. If <code>index=NULL</code> , the indices are randomly chosen.
<code>path</code>	Bootstrap coefficients for either <code>glm path</code> or <code>cox path</code> are computed. Default is <code>path=glm path</code> .
<code>method</code>	For each bootstrap run, the regularization parameter is determined based on either <code>aic</code> or <code>bic</code> . Default is <code>aic</code> .
<code>trace</code>	If <code>TRUE</code> , the number of bootstrap runs is printed out.
<code>...</code>	other options for <code>glm path</code> or <code>cox path</code>

Details

Fitting `glm path` or `cox path` gives a series of solution sets with a varying size of the active set. Once we select an appropriate value of the regularization parameter, and thus a set of coefficients, we may then validate the chosen coefficients through bootstrap analysis. `plot.bootstrap` summarizes the bootstrap results by generating the histograms or the pair scatter plots of the bootstrap coefficients.

Value

`bootstrap.path` returns a `bootpath` object, which is a matrix (`B` by `ncol(x)`) of bootstrap coefficients. Coefficients computed from the whole data are stored as an attribute `coefficients`.

Author(s)

Mee Young Park and Trevor Hastie

References

Bradley Efron and Robert Tibshirani (1993) *An Introduction to the Bootstrap* CHAPMAN & HALL/CRC, Boca Raton.

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

coxpath, glmfpath, plot.bootstrap

Examples

```
data(heart.data)
attach(heart.data)
bootstrap.a <- bootstrap.path(x, y, B=100, method="bic")
detach(heart.data)
data(lung.data)
attach(lung.data)
bootstrap.b <- bootstrap.path(data=lung.data, B=100, path="coxpath")
detach(lung.data)
```

coxpath

Fits the entire L1 regularization path for Cox proportional hazards model

Description

This algorithm uses predictor-corrector method to compute the entire regularization path for Cox proportional hazards model with L1 penalty.

Usage

```
coxpath(data, nopenalty.subset = NULL, method = c("breslow", "efron"),
        lambda2 = 1e-5, max.steps = 10*min(n, m), max.norm = 100*m,
        min.lambda = (if (m >= n) 1e-3 else 0), max.vars = Inf,
        max.arclength = Inf, frac.arclength = 1, add.newvars = 1,
        bshoot.threshold = 0.1, relax.lambda = 1e-7,
        approx.Gram = FALSE, standardize = TRUE,
        function.precision = 3e-13, eps = .Machine$double.eps,
        trace = FALSE)
```

Arguments

<code>data</code>	a list consisting of <code>x</code> : a matrix of features, <code>time</code> : the survival time, and <code>status</code> : censor status with 1 if died and 0 if censored.
<code>nopenalty.subset</code>	a set of indices for the predictors that are not subject to the L1 penalty
<code>method</code>	approximation method for tied survival times. Approximations derived by Breslow (1974) and Efron (1977) are available. Default is <code>breslow</code> .
<code>lambda2</code>	regularization parameter for the L2 norm of the coefficients. Default is $1e-5$.
<code>max.steps</code>	an optional bound for the number of steps to be taken. Default is $10 * \min\{\text{nrow}(x), \text{ncol}(x)\}$.
<code>max.norm</code>	an optional bound for the L1 norm of the coefficients. Default is $100 * \text{ncol}(x)$.
<code>min.lambda</code>	an optional (lower) bound for the size of λ . When $\text{ncol}(x)$ is relatively large, the coefficient estimates are prone to numerical precision errors at extremely small λ . In such cases, early stopping is recommended. Default is 0 for $\text{ncol}(x) < \text{nrow}(x)$ cases and $1e-3$ otherwise.
<code>max.vars</code>	an optional bound for the number of active variables. Default is <code>Inf</code> .
<code>max.arclength</code>	an optional bound for arc length (L1 norm) of a step. If <code>max.arclength</code> is extremely small, an exact nonlinear path is produced. Default is <code>Inf</code> .
<code>frac.arclength</code>	Under the default setting, the next step size is computed so that the active set changes right at the next value of <code>lambda</code> . When <code>frac.arclength</code> is assigned some fraction between 0 and 1, the step size is decreased by the factor of <code>frac.arclength</code> in arc length. If <code>frac.arclength=0.2</code> , the step length is adjusted so that the active set would change after five smaller steps. Either <code>max.arclength</code> or <code>frac.arclength</code> can be used to force the path to be more accurate. Default is 1.
<code>add.newvars</code>	<code>add.newvars</code> candidate variables (that are currently not in the active set) are used in the corrector step as potential active variables. Default is 1.
<code>bshoot.threshold</code>	If the absolute value of a coefficient is larger than <code>bshoot.threshold</code> at the first corrector step it becomes nonzero (therefore when λ is considered to have been decreased too far), λ is increased again. i.e. A backward distance in λ that makes the coefficient zero is computed. Default is 0.1.
<code>relax.lambda</code>	A variable joins the active set if $ l'(\beta) > \lambda * (1 - \text{relax.lambda})$. Default is $1e-7$. If no variable joins the active set even after many (>20) steps, the user should increase <code>relax.lambda</code> to $1e-6$ or $1e-5$, but not more than that. This adjustment is sometimes needed because of the numerical precision/error propagation problems. In general, the paths are less accurate with relaxed <code>lambda</code> .
<code>approx.Gram</code>	If <code>TRUE</code> , an approximated Gram matrix is used in predictor steps; each step takes less number of computations, but the total number of steps usually increases. This might be useful when the number of features is large.
<code>standardize</code>	If <code>TRUE</code> , predictors are standardized to have a unit variance.

<code>function.precision</code>	<code>function.precision</code> parameter used in the internal solver. Default is $3e-13$. The algorithm is faster, but less accurate with relaxed, larger <code>function.precision</code> .
<code>eps</code>	an effective zero
<code>trace</code>	If TRUE, the algorithm prints out its progress.

Details

This algorithm implements the predictor-corrector method to determine the entire path of the coefficient estimates as the amount of regularization varies; it computes a series of solution sets, each time estimating the coefficients with less regularization, based on the previous estimate. The coefficients are estimated with no error at the knots, and the values are connected, thereby making the paths piecewise linear.

We thank Michael Saunders of SOL, Stanford University for providing the solver used for the convex optimization in corrector steps of `coxpath`.

Value

A `coxpath` object is returned.

<code>lambda</code>	vector of λ values for which the exact coefficients are computed
<code>lambda2</code>	λ_2 used
<code>step.length</code>	vector of step lengths in λ
<code>corr</code>	matrix of $l'(\beta)$ values (derivatives of the log-partial-likelihood)
<code>new.df</code>	vector of degrees of freedom (to be used in the plot function)
<code>df</code>	vector of degrees of freedom at each step
<code>loglik</code>	vector of log-partial-likelihood computed at each step
<code>aic</code>	vector of AIC values
<code>bic</code>	vector of BIC values
<code>b.predictor</code>	matrix of coefficient estimates from the predictor steps
<code>b.corrector</code>	matrix of coefficient estimates from the corrector steps
<code>new.A</code>	vector of boolean values indicating the steps at which the active set changed (to be used in the plot/predict functions)
<code>actions</code>	actions taken at each step
<code>meanx</code>	means of the columns of x
<code>sdx</code>	standard deviations of the columns of x
<code>xnames</code>	column names of x
<code>method</code>	method used
<code>nopenalty.subset</code>	<code>nopenalty.subset</code> used
<code>standardize</code>	TRUE if the predictors were standardized before fitting

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

cv.coxpath, plot.coxpath, predict.coxpath, summary.coxpath

Examples

```
data(lung.data)
attach(lung.data)
fit.a <- coxpath(lung.data)
fit.b <- coxpath(lung.data, method="efron")
detach(lung.data)
```

cv.coxpath

Computes cross-validated (minus) log-partial-likelihoods for coxpath

Description

This function computes cross-validated (minus) log-partial-likelihoods for `coxpath`.

Usage

```
cv.coxpath(data, method = c("breslow", "efron"), nfold = 5,
           fraction = seq(from=0, to=1, length=100),
           mode = c("norm", "lambda"), plot.it = TRUE, se = TRUE, ...)
```

Arguments

<code>data</code>	a list consisting of <code>x</code> : a matrix of features, <code>time</code> : the survival time, and <code>status</code> : censor status with 1 if died and 0 if censored.
<code>method</code>	approximation method for tied survival times. Approximations derived by Breslow (1974) and Efron (1977) are available. Default is <code>breslow</code> .
<code>nfold</code>	number of folds to be used in cross-validation. Default is <code>nfold=5</code> .
<code>fraction</code>	the fraction of L1 norm or $\log(\lambda)$ with respect to their maximum values at which the CV errors are computed. Default is <code>seq(0, 1, length=100)</code> .
<code>mode</code>	If <code>mode=norm</code> , cross-validation is run at certain values of L1 norm. If <code>mode=lambda</code> , cross-validation is run at certain values of $\log(\lambda)$. Default is <code>norm</code> .
<code>plot.it</code>	If <code>TRUE</code> , CV curve is plotted.
<code>se</code>	If <code>TRUE</code> , standard errors are plotted.
<code>...</code>	other options for <code>coxpath</code>

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

coxpath, plot.coxpath, predict.coxpath

Examples

```
data(lung.data)
attach(lung.data)
cv <- cv.coxpath(lung.data)
detach(lung.data)
```

cv.glmpath

Computes cross-validated (minus) log-likelihoods or prediction errors for glmpath

Description

This function computes cross-validated (minus) log-likelihoods or prediction errors for `glmpath`.

Usage

```
cv.glmpath(x, y, data, family = binomial, weight = rep(1, n),
           offset = rep(0, n), nfold = 10,
           fraction = seq(from=0, to=1, length=100),
           type = c("loglik", "response"), mode = c("norm", "lambda"),
           plot.it = TRUE, se = TRUE, ...)
```

Arguments

x	matrix of features
y	response
data	a list consisting of <code>x</code> : a matrix of features and <code>y</code> : response. <code>data</code> is not needed if above <code>x</code> and <code>y</code> are input separately.
family	name of a family function that represents the distribution of <code>y</code> to be used in the model. It must be <code>binomial</code> , <code>gaussian</code> , or <code>poisson</code> . For each one, the canonical link function is used; <code>logit</code> for binomial, <code>identity</code> for gaussian, and <code>log</code> for poisson distribution. Default is <code>binomial</code> .
weight	an optional vector of weights for observations

<code>offset</code>	an optional vector of offset. If a column of <code>x</code> is used as <code>offset</code> , the corresponding column must be excluded from <code>x</code> .
<code>nfold</code>	number of folds to be used in cross-validation. Default is <code>nfold=10</code> .
<code>fraction</code>	the fraction of L1 norm or $\log(\lambda)$ with respect to their maximum values at which the CV errors are computed. Default is <code>seq(0, 1, length=100)</code> .
<code>type</code>	If <code>type=loglik</code> , cross-validated minus log-likelihoods are computed. If <code>type=response</code> , cross-validated prediction errors are computed. Default is <code>loglik</code> .
<code>mode</code>	If <code>mode=norm</code> , cross-validation is run at certain values of L1 norm. If <code>mode=lambda</code> , cross-validation is run at certain values of $\log(\lambda)$. Default is <code>norm</code> .
<code>plot.it</code>	If <code>TRUE</code> , CV curve is plotted.
<code>se</code>	If <code>TRUE</code> , standard errors are plotted.
<code>...</code>	other options for <code>glmpath</code>

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

`glmpath`, `plot.glmpath`, `predict.glmpath`

Examples

```
data(heart.data)
attach(heart.data)
cv.a <- cv.glmpath(x, y, family=binomial)
cv.b <- cv.glmpath(x, y, family=binomial, type="response")
detach(heart.data)
```

`glmpath`

Fits the entire L1 regularization path for generalized linear models

Description

This algorithm uses predictor-corrector method to compute the entire regularization path for generalized linear models with L1 penalty.

Usage

```
glmpath(x, y, data, nopenalty.subset = NULL, family = binomial,
        weight = rep(1, n), offset = rep(0, n), lambda2 = 1e-5,
        max.steps = 10*min(n, m), max.norm = 100*m,
        min.lambda = (if (m >= n) 1e-6 else 0), max.vars = Inf,
        max.arclength = Inf, frac.arclength = 1, add.newvars = 1,
        bshoot.threshold = 0.1, relax.lambda = 1e-8,
        standardize = TRUE, function.precision = 3e-13,
        eps = .Machine$double.eps, trace = FALSE)
```

Arguments

<code>x</code>	matrix of features
<code>y</code>	response
<code>data</code>	a list consisting of <code>x</code> : a matrix of features and <code>y</code> : response. <code>data</code> is not needed if <code>x</code> and <code>y</code> are input separately.
<code>nopenalty.subset</code>	a set of indices for the predictors that are not subject to the L1 penalty
<code>family</code>	name of a family function that represents the distribution of <code>y</code> to be used in the model. It must be <code>binomial</code> , <code>gaussian</code> , or <code>poisson</code> . For each one, the canonical link function is used; <code>logit</code> for binomial, <code>identity</code> for gaussian, and <code>log</code> for poisson distribution. Default is <code>binomial</code> .
<code>weight</code>	an optional vector of weights for observations
<code>offset</code>	an optional vector of offset. If a column of <code>x</code> is used as offset, the corresponding column must be removed from <code>x</code> .
<code>lambda2</code>	regularization parameter for the L2 norm of the coefficients. Default is <code>1e-5</code> .
<code>max.steps</code>	an optional bound for the number of steps to be taken. Default is <code>10 * min{nrow(x), ncol(x)}</code> .
<code>max.norm</code>	an optional bound for the L1 norm of the coefficients. Default is <code>100 * ncol(x)</code> .
<code>min.lambda</code>	an optional (lower) bound for the size of λ . Default is <code>0</code> for <code>ncol(x) < nrow(x)</code> cases and <code>1e-6</code> otherwise.
<code>max.vars</code>	an optional bound for the number of active variables. Default is <code>Inf</code> .
<code>max.arclength</code>	an optional bound for arc length (L1 norm) of a step. If <code>max.arclength</code> is extremely small, an exact nonlinear path is produced. Default is <code>Inf</code> .
<code>frac.arclength</code>	Under the default setting, the next step size is computed so that the active set changes right at the next value of <code>lambda</code> . When <code>frac.arclength</code> is assigned some fraction between 0 and 1, the step size is decreased by the factor of <code>frac.arclength</code> in arc length. If <code>frac.arclength=0.2</code> , the step length is adjusted so that the active set would change after five smaller steps. Either <code>max.arclength</code> or <code>frac.arclength</code> can be used to force the path to be more accurate. Default is <code>1</code> .
<code>add.newvars</code>	<code>add.newvars</code> candidate variables (that are currently not in the active set) are used in the corrector step as potential active variables. Default is <code>1</code> .

<code>bshoot.threshold</code>	If the absolute value of a coefficient is larger than <code>bshoot.threshold</code> at the first corrector step it becomes nonzero (therefore when λ is considered to have been decreased too far), λ is increased again. i.e. A backward distance in λ that makes the coefficient zero is computed. Default is 0.1.
<code>relax.lambda</code>	A variable joins the active set if $ l'(\beta) > \lambda * (1 - \text{relax.lambda})$. Default is $1e-8$. If no variable joins the active set even after many (>20) steps, the user should increase <code>relax.lambda</code> to $1e-7$ or $1e-6$, but not more than that. This adjustment is sometimes needed because of the numerical precision/error propagation problems. In general, the paths are less accurate with relaxed lambda.
<code>standardize</code>	If TRUE, predictors are standardized to have a unit variance.
<code>function.precision</code>	<code>function.precision</code> parameter used in the internal solver. Default is $3e-13$. The algorithm is faster, but less accurate with relaxed, larger function precision.
<code>eps</code>	an effective zero
<code>trace</code>	If TRUE, the algorithm prints out its progress.

Details

This algorithm implements the predictor-corrector method to determine the entire path of the coefficient estimates as the amount of regularization varies; it computes a series of solution sets, each time estimating the coefficients with less regularization, based on the previous estimate. The coefficients are estimated with no error at the knots, and the values are connected, thereby making the paths piecewise linear.

We thank Michael Saunders of SOL, Stanford University for providing the solver used for the convex optimization in corrector steps of `glmpath`.

Value

A `glmpath` object is returned.

<code>lambda</code>	vector of λ values for which the exact coefficients are computed
<code>lambda2</code>	λ_2 used
<code>step.length</code>	vector of step lengths in λ
<code>corr</code>	matrix of $l'(\beta)$ values (derivatives of the log-likelihood)
<code>new.df</code>	vector of degrees of freedom (to be used in the plot function)
<code>df</code>	vector of degrees of freedom at each step
<code>deviance</code>	vector of deviance computed at each step
<code>aic</code>	vector of AIC values
<code>bic</code>	vector of BIC values
<code>b.predictor</code>	matrix of coefficient estimates from the predictor steps
<code>b.corrector</code>	matrix of coefficient estimates from the corrector steps

new.A	vector of boolean values indicating the steps at which the active set changed (to be used in the plot/predict functions)
actions	actions taken at each step
meanx	means of the columns of x
sdx	standard deviations of the columns of x
xnames	column names of x
family	family used
weight	weights used
offset	offset used
nopenalty.subset	nopenalty.subset used
standardize	TRUE if the predictors were standardized before fitting

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

cv.glmpath, plot.glmpath, predict.glmpath, summary.glmpath

Examples

```
data(heart.data)
attach(heart.data)
fit.a <- glmpath(x, y, family=binomial)
fit.b <- glmpath(x, y, family=gaussian)
detach(heart.data)
```

heart.data

Dataset for glmpath

Description

South African Heart Disease dataset used to test glmpath algorithm

Usage

```
data(heart.data)
```

Format

A dataset with 462 observations on 9 variables and a binary response.

x *x* contains 9 columns of the following variables: sbp (systolic blood pressure); tobacco (cumulative tobacco); ldl (low density lipoprotein cholesterol); adiposity; famhist (family history of heart disease); typea (type-A behavior); obesity; alcohol (current alcohol consumption); age (age at onset)

y response, coronary heart disease

References

Hastie, T., Tibshirani, R., and Friedman, J. (2001) *Elements of Statistical Learning; Data Mining, Inference, and Prediction* Springer-Verlag, New York.

Examples

```
data(heart.data)
attach(heart.data)
fit <- glm(x, y, family=binomial)
detach(heart.data)
```

lung.data

Dataset for coxpath

Description

Lung cancer dataset used to test `coxpath` algorithm

Usage

```
data(lung.data)
```

Format

A dataset consisting of 137 observations with their survival time, censor status as well as 6 features.

x *x* contains 6 columns of the following variables: trt (1=standard treatment, and 2=test); celltype (1=squamous, 2=smallcell, 3=adeno, and 4=large); karno (Karnofsky performance score); diagtime (months from diagnosis to randomization); age (in years); prior (prior therapy 0=no, and 1=yes)

time survival time

status censor status

References

Kalbfleisch, J. and Prentice, R. (2002) *The Statistical Analysis of Failure Time Data* J. Wiley, Hoboken, N.J.

Examples

```
data(lung.data)
attach(lung.data)
fit <- coxpath(lung.data)
detach(lung.data)
```

plot.bootpath	<i>Generates the histograms or the pairwise scatter plots of the bootstrap coefficients computed from bootstrap.path</i>
---------------	--

Description

This function takes a `bootpath` object from `bootstrap.path` and generates the histograms or the pairwise scatter plots of the bootstrap coefficients.

Usage

```
plot.bootpath(x, type=c("histogram", "pairplot"),
              mfrow = NULL, mar = NULL, ...)
```

Arguments

<code>x</code>	a <code>bootpath</code> object from <code>bootstrap.path</code> .
<code>type</code>	If <code>type=histogram</code> , the histograms of bootstrap coefficients for individual features are generated. The red vertical bar indicates the coefficient computed using the whole data. The thick bar at zero indicates the frequency of the zero coefficients. If <code>type=pairplot</code> , the pairwise scatter plots of the bootstrap coefficients are generated. The red solid dot indicates the pair of coefficients computed using the whole data. Default is <code>histogram</code> .
<code>mfrow</code>	determines the numbers of rows and columns of the histograms on a page. 2 rows are generated as a default.
<code>mar</code>	margin relative to the current font size
<code>...</code>	other options for the plot

Details

Fitting `glm` or `cox` gives a series of solution sets with a varying size of the active set. Once we select an appropriate value of the regularization parameter, and, thus a set of coefficients, we may then validate the chosen coefficients through a bootstrap analysis. `plot.bootstrap` summarizes the bootstrap results by generating the histograms or the pairwise scatter plots of the bootstrap coefficients.

Author(s)

Mee Young Park and Trevor Hastie

References

Bradley Efron and Robert Tibshirani (1993) *An Introduction to the Bootstrap* CHAPMAN & HALL/CRC, Boca Raton.

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

bootstrap.path, coxpath, glmpath

Examples

```
data(heart.data)
attach(heart.data)
bootstrap.a <- bootstrap.path(x, y, B=100)
plot(bootstrap.a)
plot(bootstrap.a, type="pairplot")
detach(heart.data)
```

plot.coxpath

Plots the regularization path computed from coxpath

Description

This function takes a `coxpath` object and visualizes the regularization path. The horizontal axis can be `norm`, `lambda` or `step`. The vertical axis can be `coefficients`, `aic` or `bic`.

Usage

```
plot.coxpath(x, xvar = c("norm", "lambda", "step"),
             type = c("coefficients", "aic", "bic"),
             plot.all.steps = FALSE, xlimit = NULL, predictor = FALSE,
             omit.zero = TRUE, breaks = TRUE, mar = NULL, main = NULL,
             eps = .Machine$double.eps, ...)
```

Arguments

<code>x</code>	a <code>coxpath</code> object
<code>xvar</code>	horizontal axis. <code>xvar=norm</code> plots against the L1 norm of the coefficients (to which L1 norm penalty was applied); <code>xvar=lambda</code> plots against λ ; and <code>xvar=step</code> plots against the number of steps taken. Default is <code>norm</code> .
<code>type</code>	type of the plot, or the vertical axis. Default is <code>coefficients</code> .
<code>plot.all.steps</code>	If <code>TRUE</code> , all the steps taken along the path are marked on the plot. If <code>FALSE</code> , which is the default, only the steps at which the active set changed are shown on the plot.

xlimit	When the user wants to visualize a (beginning) sub-part of the plot, xlimit sets an upper limit to the L1 norm or the number of steps, or a lower limit to λ .
predictor	If TRUE and type=coefficients, the predictor step estimates are connected with dotted lines. If FALSE, only the corrector step estimates are connected with solid lines.
omit.zero	If TRUE, the predictors that were never in the active set are omitted.
breaks	If TRUE, vertical lines are drawn at the points where the active set changes and numbered with the degrees of freedom.
mar	margin relative to the current font size
main	title of the plot
eps	an effective zero
...	other options for the plot

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

cv.coxpath, coxpath, predict.coxpath

Examples

```
data(lung.data)
attach(lung.data)
fit <- coxpath(lung.data)
par(mfrow=c(3,2))
plot(fit)
plot(fit,xvar="lambda")
plot(fit,xvar="step")
plot(fit,xvar="step",xlimit=8)
plot(fit,type="aic")
plot(fit,type="bic")
detach(lung.data)
```

plot.glmpath *Plots the regularization path computed from glmpath*

Description

This function takes a `glmpath` object and visualizes the regularization path. The horizontal axis can be `norm`, `lambda` or `step`. The vertical axis can be `coefficients`, `aic` or `bic`.

Usage

```
plot.glmpath(x, xvar = c("norm", "lambda", "step"),
             type = c("coefficients", "aic", "bic"),
             plot.all.steps = FALSE, xlimit = NULL, predictor = FALSE,
             omit.zero = TRUE, breaks = TRUE, mar = NULL,
             eps = .Machine$double.eps, main = NULL, ...)
```

Arguments

<code>x</code>	a <code>glmpath</code> object
<code>xvar</code>	horizontal axis. <code>xvar=norm</code> plots against the L1 norm of the coefficients (to which L1 norm penalty was applied); <code>xvar=lambda</code> plots against λ ; and <code>xvar=step</code> plots against the number of steps taken. Default is <code>norm</code> .
<code>type</code>	type of the plot, or the vertical axis. Default is <code>coefficients</code> .
<code>plot.all.steps</code>	If <code>TRUE</code> , all the steps taken along the path are marked on the plot. If <code>FALSE</code> , which is the default, only the steps at which the active set changed are shown on the plot.
<code>xlimit</code>	When the user wants to visualize a (beginning) sub-part of the plot, <code>xlimit</code> sets an upper limit to the L1 norm or the number of steps, or a lower limit to λ .
<code>predictor</code>	If <code>TRUE</code> and <code>type=coefficients</code> , the predictor step estimates are connected with dotted lines. If <code>FALSE</code> , only the corrector step estimates are connected with solid lines.
<code>omit.zero</code>	If <code>TRUE</code> and <code>type=coefficients</code> , the predictors that were never in the active set are omitted.
<code>breaks</code>	If <code>TRUE</code> , vertical lines are drawn at the points where the active set changes and numbered with the degrees of freedom.
<code>mar</code>	margin relative to the current font size
<code>eps</code>	an effective zero
<code>main</code>	title of the plot
<code>...</code>	other options for the plot

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

cv.glmpath, glmpath, predict.glmpath

Examples

```
data(heart.data)
attach(heart.data)
fit <- glmpath(x, y, family=binomial)
par(mfrow=c(3,2))
plot(fit)
plot(fit,xvar="lambda")
plot(fit,xvar="step")
plot(fit,xvar="step",xlimit=8)
plot(fit,type="aic")
plot(fit,type="bic")
detach(heart.data)
```

predict.coxpath *Makes predictions at particular points along the fitted coxpath*

Description

This function makes predictions at particular points along the fitted `coxpath`. The coefficients, log-partial-likelihood, linear predictor or the risk can be computed. A `coxph` object can be returned at one particular value of λ .

Usage

```
predict.coxpath(object, data, s, type = c("coefficients", "loglik",
    "lp", "risk", "coxph"), mode = c("step",
    "norm.fraction", "norm", "lambda.fraction", "lambda"),
    eps = .Machine$double.eps, ...)
```

Arguments

<code>object</code>	a <code>coxpath</code> object
<code>data</code>	a list containing <code>x</code> , <code>time</code> , and <code>status</code> , with which the predictions are made. If <code>type=lp</code> or <code>type=risk</code> , then <code>x</code> is required. If <code>type=loglik</code> or <code>type=coxph</code> , then <code>x</code> , <code>time</code> , and <code>status</code> are required.
<code>s</code>	the values of <code>mode</code> at which the predictions are made. If <code>type=coxph</code> , only the first element of <code>s</code> is used. If <code>s</code> is missing, then the steps at which the active set changed are used, and thus, <code>mode</code> is automatically switched to <code>step</code> .

type	If <code>type=coefficients</code> , the coefficients are returned; if <code>type=loglik</code> , log-partial-likelihoods are returned; if <code>type=lp</code> , linear predictors ($x'\beta$) are returned; if <code>type=risk</code> , risks ($e^{x'\beta}$) are returned; and if <code>type=coxph</code> , a <code>coxph</code> object (as in survival package) at the first element of <code>s</code> is returned. (i.e. the components of a <code>coxph</code> object such as coefficients, variance, and the test statistics are adjusted to the shrinkage corresponding to <code>s</code> . A <code>coxph</code> object can be further used as an argument to the functions in survival package.) Default is <code>coefficients</code> . The coefficients for the initial input variables are returned (rather than the standardized coefficients).
mode	what <code>mode=s</code> refers to. If <code>mode=step</code> , <code>s</code> is the number of steps taken; if <code>mode=norm.fraction</code> , <code>s</code> is the fraction of the L1 norm of the standardized coefficients (with respect to the largest norm); if <code>mode=norm</code> , <code>s</code> is the L1 norm of the standardized coefficients; if <code>mode=lambda.fraction</code> , <code>s</code> is the fraction of $\log(\lambda)$; and if <code>mode=lambda</code> , <code>s</code> is λ . Default is <code>step</code> .
eps	an effective zero
...	other options for the prediction

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

`cv.coxpath`, `coxpath`, `plot.coxpath`

Examples

```
data(lung.data)
attach(lung.data)
fit <- coxpath(lung.data)
pred.a <- predict(fit, x, s = seq(0, 1, length=10),
                 mode = "norm.fraction")
library(survival)
pred.b <- predict(fit, lung.data, s = 0.5, type = "coxph",
                 mode = "lambda.fraction")
pred.s <- survfit(pred.b)
plot(pred.s)
detach(lung.data)
```

predict.glmpath *Makes predictions at particular points along the fitted glmpath*

Description

This function makes predictions at particular points along the fitted `glmpath`. The linear predictor, estimated response, log-likelihood, or the coefficients can be computed.

Usage

```
predict.glmpath(object, newx, newy, s, type = c("link", "response",
        "loglik", "coefficients"), mode = c("step",
        "norm.fraction", "norm", "lambda.fraction", "lambda"),
        weight = NULL, offset = NULL,
        eps = .Machine$double.eps, ...)
```

Arguments

object	a <code>glmpath</code> object
newx	a matrix of features at which the predictions are made. If <code>type=link</code> , <code>type=response</code> , or <code>type=loglik</code> , <code>newx</code> is required.
newy	a vector of responses corresponding to <code>newx</code> . If <code>type=loglik</code> , <code>newy</code> is required.
s	the values of <code>mode</code> at which the predictions are made. If <code>s</code> is missing, then the steps at which the active set changed are used, and thus, <code>mode</code> is automatically switched to <code>step</code> .
type	If <code>type=link</code> , the linear predictors are returned; if <code>type=response</code> , the estimated responses are returned; if <code>type=loglik</code> , the log-likelihoods are returned, and if <code>type=coefficients</code> , the coefficients are returned. The coefficients for the initial input variables are returned (rather than the standardized coefficients). Default is <code>link</code> .
mode	what <code>mode=s</code> refers to. If <code>mode=step</code> , <code>s</code> is the number of steps taken; if <code>mode=norm.fraction</code> , <code>s</code> is the fraction of the L1 norm of the standardized coefficients (with respect to the largest norm); if <code>mode=norm</code> , <code>s</code> is the L1 norm of the standardized coefficients; if <code>mode=lambda.fraction</code> , <code>s</code> is the fraction of $\log(\lambda)$; and if <code>mode=lambda</code> , <code>s</code> is λ . Default is <code>step</code> .
weight	an optional vector of weights for observations. <code>weight</code> is effective only if <code>type=loglik</code> .
offset	If <code>offset</code> was used in <code>object</code> , <code>offset</code> must be provided for prediction, unless <code>type=coefficients</code> .
eps	an effective zero
...	other options for the prediction

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

cv.glmpath, glmpath, plot.glmpath

Examples

```
data(heart.data)
attach(heart.data)
fit <- glmpath(x, y, family=binomial)
pred <- predict(fit, x, s = seq(0, 1, length=10), mode="norm.fraction")
detach(heart.data)
```

summary.coxpath *Produces an anova-type summary for a coxpath object*

Description

This function produces an anova-type summary for a coxpath object.

Usage

```
summary.coxpath(object, ...)
```

Arguments

object	a coxpath object
...	additional arguments

Details

An anova type of summary is returned, including Df, Log-partial-likelihood, AIC, and BIC values for the steps where the active set changed.

Value

A data.frame is returned, with the following components at transition points:

Df	degrees of freedom at each step
Log.p.lik	log-partial-likelihood at each step
AIC	AIC value at each step
BIC	BIC value at each step

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

coxpath, plot.coxpath, print.coxpath

Examples

```
data(lung.data)
attach(lung.data)
fit <- coxpath(lung.data)
summary(fit)
detach(lung.data)
```

summary.glmpath *Produces an anova-type summary for a glmpath object*

Description

This function produces an anova-type summary for a glmpath object.

Usage

```
summary.glmpath(object, ...)
```

Arguments

object	a glmpath object
...	additional arguments

Details

An anova type of summary is returned, including Df, Deviance, AIC, and BIC values for the steps where the active set changed.

Value

A data.frame is returned, with the following components at transition points:

Df	degrees of freedom at each step
Deviance	deviance computed at each step
AIC	AIC value at each step
BIC	BIC value at each step

Author(s)

Mee Young Park and Trevor Hastie

References

Mee Young Park and Trevor Hastie (2007) L1 regularization path algorithm for generalized linear models. *J. R. Statist. Soc. B*, 69, 659-677.

See Also

glmpath, plot.glmpath, print.glmpath

Examples

```
data(heart.data)
attach(heart.data)
fit <- glmpath(x, y)
summary(fit)
detach(heart.data)
```

Index

*Topic **datasets**

heart.data, 11
lung.data, 12

*Topic **models**

bootstrap.path, 1
coxpath, 3
cv.coxpath, 6
cv.glmpath, 7
glmpath, 8
plot.bootpath, 12
plot.coxpath, 14
plot.glmpath, 15
predict.coxpath, 17
predict.glmpath, 18
summary.coxpath, 19
summary.glmpath, 21

*Topic **regression**

bootstrap.path, 1
coxpath, 3
cv.coxpath, 6
cv.glmpath, 7
glmpath, 8
plot.bootpath, 12
plot.coxpath, 14
plot.glmpath, 15
predict.coxpath, 17
predict.glmpath, 18
summary.coxpath, 19
summary.glmpath, 21

bootstrap.path, 1

coxpath, 3
cv.coxpath, 6
cv.glmpath, 7

glmpath, 8

heart.data, 11

lung.data, 12

plot.bootpath, 12
plot.coxpath, 14
plot.glmpath, 15
predict.coxpath, 17
predict.glmpath, 18

summary.coxpath, 19
summary.glmpath, 21