Package 'glmnet'

April 17, 2009

Type Package

Title Lasso and elastic-net regularized generalized linear models

Version 1.1-3

Date 2009-01-22

Author Jerome Friedman, Trevor Hastie, Rob Tibshirani

Maintainer Trevor Hastie <hastie@stanford.edu>

Depends Matrix

Description Extremely efficient procedures for fitting the entire lasso or elastic-net regularization path for linear regression, logistic and multinomial regression models. The algorithm uses cyclical coordinate descent in a pathwise fashion, as described in the paper on the maintainer's website.

License GPL-2

URL http://www-stat.stanford.edu/~hastie/Papers/glmnet.pdf

Repository CRAN

Date/Publication 2009-01-24 10:12:48

R topics documented:

`glmnet-package` *Elastic net model paths for some generalized linear models*

Description

This package fits lasso and elastic-net model paths for regression, logistic and multinomial regression using coordinate descent. The algorithm is extremely fast, and exploits sparsity in the input x matrix where it exists. A variety of predictions can be made from the fitted models.

Details

Package:	glmnet
Type:	Package
Version:	1.0
Date:	2008-05-14
License:	What license is it under?

Very simple to use. Accepts `x,y` data for regression models, and produces the regularization path over a grid of values for the tuning parameter `lambda`. Only 5 functions: `glmnet`
`predict.glmnet`
`plot.glmnet`
`print.glmnet`
`coef.glmnet`

Author(s)

Jerome Friedman, Trevor Hastie and Rob Tibshirani
Maintainer: Trevor Hastie <hastie@stanford.edu>

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) *Regularization Paths for Generalized Linear Models via Coordinate Descent*

Examples

```
x=matrix(rnorm(100*20),100,20)
y=rnorm(100)
g2=sample(1:2,100,replace=TRUE)
g4=sample(1:4,100,replace=TRUE)
fit1=glmnet(x,y)
predict(fit1,newx=x[1:5,],s=c(0.01,0.005))
predict(fit1,type="coef")
plot(fit1,xvar="lambda")
fit2=glmnet(x,g2,family="binomial")
predict(fit2,type="response",newx=x[2:5,])
predict(fit2,type="nonzero")
```

```
fit3=glmnet(x,g4,family="multinomial")
predict(fit3,newx=x[1:3,],type="response",s=0.01)
```

`glmnet` *fit an elasticnet model path*

Description

Fit a regularization path for the elasticnet at a grid of values for the regularization parameter lambda. Can deal with all shapes of data, including very large sparse data matrices. Fits linear, logistic and multinomial regression models.

Usage

```
glmnet(x, y, family=c("gaussian","binomial","multinomial"), weights, alpha = 1,
  nlambda = 100, lambda.min = ifelse(nobs<nvars,0.05,0.0001), lambda,
  standardize = TRUE, thresh = 1e-04,  dfmax = nvars + 1,
  pmax = min(dfmax * 1.2, nvars), exclude, penalty.factor = rep(1, nvars),
  maxit=100, HessianExact = FALSE, type = c("covariance", "naive"))
```

Arguments

`x`	input matrix, of dimension nobs x nvars; each row is an observation vector. Can be in sparse column format (class `"dgCMatrix"` as in package `Matrix`)
`y`	response variable. Quantitative for `family="gaussian"`. For `family="binomial"` should be either a factor with two levels, or a two-column matrix of counts or proportions. For `family="multinomial"`, can be a `nc>=2` level factor, or a matrix with `nc` columns of counts or proportions
`family`	Response type (see above)
`weights`	observation weights. Can be total counts if responses are proportion matrices. Default is 1 for each observation
`alpha`	The elasticnet mixing parameter, with $0 < \alpha \leq 1$. The penalty is defined as $$(1-\alpha)/2\|\|\beta\|\|_2^2 + \alpha\|\|\beta\|\|_1.$$ `alpha=1` is the lasso penalty; Currently `alpha<0.01` not reliable, unless you supply your own `lambda` sequence
`nlambda`	The number of `lambda` values - default is 100.
`lambda.min`	Smallest value for `lambda`, as a fraction of `lambda.max`, the (data derived) entry value (i.e. the smallest value for which all coefficients are zero). The default depends on the sample size `nobs` relative to the number of variables `nvars`. If `nobs > nvars`, the default is `0.0001`, close to zero. If `nobs < nvars`, the default is `0.05`. A very small value of `lambda.min` will lead to a saturated fit. This is undefined for `"binomial"` and `"multinomial"` models, and `glmnet` will exit gracefully when the percentage deviance explained is almost 1.

`lambda` A user supplied `lambda` sequence. Typical usage is to have the program compute its own `lambda` sequence based on `nlambda` and `lambda.min`. Supplying a value of `lambda` overrides this. Use with care - it is better to supply a decreasing sequence of `lambda` values than a single (small) value.

`standardize` Logical flag for variable standardization, prior to fitting the model sequence. The coefficients are always returned on the original scale. Default is is `standardize=TRUE`

`thresh` Convergence threshold for coordinate descent. Each inner coordinate-descent loop continues until the relative change in any coefficient is less than `thresh`. Defaults value is `1E-4`.

`dfmax` Limit the maximum number of variables in the model. Useful for very large `nvars`, if a partial path is desired.

`pmax` Limit the maximum number of variables ever to be nonzero

`exclude` Indices of variables to be excluded from the model. Default is none. Equivalent to an infinite penalty factor (next item).

`penalty.factor` Separate penalty factors can be applied to each coefficient. This is a number that multiplies `lambda` to allow differential shrinkage. Can be 0 for some variables, which implies no shrinkage, and that variable is always included in the model. Default is 1 for all variables (and implicitly infinity for variables listed in `exclude`).

`maxit` Maximum number of outer-loop iterations for `"binomial"` or `"multinomial"` families. Default is 100.

`HessianExact` Only applies to `"binomial"` or `"multinomial"` families. If `FALSE` (the default), an upper-bound approximation is made to the hessian, which is not recalculated at each outer loop.

`type` Two algorithm types are supported for (only) `family="gaussian"`. The default `type="covariance"` saves all inner-products ever computed, and can be much faster than `type="naive"`. The latter can be more efficient for `p >> N` situations.

Details

The sequence of models implied by `lambda` is fit by coordinate descent. For `family="gaussian"` this is the lasso sequence if `alpha=1`, else it is the elasticnet sequence. For `family="binomial"` or `family="multinomial"`, this is a lasso or elasticnet regularization path for fitting the linear logistic or multinomial logistic regression paths. Sometimes the sequence is truncated before `nlambda` values of `lambda` have been used, because of instabilities in the logistic or multinomial models near a saturated fit. `glmnet(...,family="binomial")` fits a traditional logistic regression model for the log-odds. `glmnet(...,family="multinomial")` fits a symmetric multinomial model, where each class is represented by a linear model (on the log-scale). The penalties take care of redundancies. A two-class `"multinomial"` model will produce the same fit as the corresponding `"binomial"` model, except the pair of coefficient matrices will be equal in magnitude and opposite in sign, and half the `"binomial"` values. Note that the objective function for `"gaussian"` is

$$1/(2 * nobs)RSS + \lambda * penalty$$

, and for the logistic models it is

$$1/nobs - loglik + \lambda * penalty$$

Value

An object with S3 class `"glmnet","*"` , where `"*"` is `"elnet"`, `"lognet"` or `"multnet"` for the three types of models.

`call`	the call that produced this object
`a0`	Intercept sequence of length `length(lambda)`
`beta`	For `"elnet"` and `"lognet"` models, a `nvars x length(lambda)` matrix of coefficients, stored in sparse column format (`"dgCMatrix"`). For `"multnet"`, a list of `nc` such matrices, one for each class.
`lambda`	The actual sequence of `lambda` values used
`dev`	The fraction of (null) deviance explained (for `"elnet"`, this is the R-square).
`nulldev`	Null deviance (per observation)
`df`	The number of nonzero coefficients for each value of `lambda`. For `"multnet"`, this is the number of variables with a nonzero coefficient for *any* class.
`dfmat`	For `"multnet"` only. A matrix consisting of the number of nonzero coefficients per class
`dim`	dimension of coefficient matrix (ices)
`npasses`	total passes over the data summed over all lambda values
`jerr`	error flag, for warnings and errors (largely for internal debugging).

Author(s)

Jerome Friedman, Trevor Hastie and Rob Tibshirani
Maintainer: Trevor Hastie ⟨hastie@stanford.edu⟩

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) *Regularization Paths for Generalized Linear Models via Coordinate Descent*http://www-stat.stanford.edu/~hastie/Papers/glmnet.pdf

See Also

`print`, `predict` and `coef` methods.

Examples

```
x=matrix(rnorm(100*20),100,20)
y=rnorm(100)
g2=sample(1:2,100,replace=TRUE)
g4=sample(1:4,100,replace=TRUE)
fit1=glmnet(x,y)
print(fit1)
```

```
coef(fit1,s=0.01) # extract coefficients at a single value of lambda
predict(fit1,newx=x[1:10,],s=c(0.01,0.005)) # make predictions
fit2=glmnet(x,g2,family="binomial")
fit3=glmnet(x,g4,family="multinomial")
```

`plot.glmnet` *plot coefficients from a "glmnet" object*

Description

Produces a coefficient profile plot of the coefficient paths for a fitted `"glmnet"` object.

Usage

```
## S3 method for class 'glmnet':
plot(x, xvar = c("norm", "lambda", "dev"), label = FALSE, ...)
```

Arguments

`x`	fitted `"glmnet"` model
`xvar`	What is on the X-axis. `"norm"` plots against the L1-norm of the coefficients, `"lambda"` against the log-lambda sequence, and `"dev"` against the percent deviance explained.
`label`	If `TRUE`, label the curves with variable sequence numbers.
`...`	Other graphical parameters to plot

Details

A coefficient profile plot is produced. If `x` is a multinomial model, a coefficient plot is produced for each class.

Author(s)

Jerome Friedman, Trevor Hastie and Rob Tibshirani
Maintainer: Trevor Hastie <hastie@stanford.edu>

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) *Regularization Paths for Generalized Linear Models via Coordinate Descent*

See Also

`glmnet`, and `print`, `predict` and `coef` methods.

Examples

```
x=matrix(rnorm(100*20),100,20)
y=rnorm(100)
g2=sample(1:2,100,replace=TRUE)
g4=sample(1:4,100,replace=TRUE)
fit1=glmnet(x,y)
plot(fit1)
plot(fit1,xvar="lambda",label=TRUE)
fit3=glmnet(x,g4,family="multinomial")
plot(fit3,type="s",pch=19)
```

---

predict.glmnet *make predictions from a "glmnet" object.*

Description

Similar to other predict methods, this functions predicts fitted values, logits, coefficients and more from a fitted "glmnet" object.

Usage

```
## S3 method for class 'glmnet':
predict(object, newx, s = object$lambda,
type=c("link","response","coefficients","class","nonzero"), exact =
FALSE, ...)
## S3 method for class 'glmnet':
coef(object,s=object$lambda, exact=FALSE, ...)
```

Arguments

object	Fitted "glmnet" model object.
newx	Matrix of new values for x at which predictions are to be made. Must be a matrix; can be sparse as in Matrix package. This argument is not used for type=c("coefficients","nonzero")
s	Value(s) of the penalty parameter lambda at which predictions are required. Default is the entire sequence used to create the model.
type	Type of prediction required. Type "link" gives the linear predictors for "binomial" or "multinomial" models; for "gaussian" models it gives the fitted values. Type "response" gives the fitted probabilities for "binomial" or "multinomial"; for "gaussian" type "response" is equivalent to type "link". Type "coefficients" computes the coefficients at the requested values for s. Note that for "binomial" models, results are returned only for the class corresponding to the second level of the factor response. Type "class" applies only to "binomial" or "multinomial" models, and produces the class label corresponding to the maximum probability. Type "nonzero" returns a list of the indices of the nonzero coefficients for each value of s.

`exact` By default (`exact=FALSE`) the predict function uses linear interpolation to make predictions for values of `s` that do not coincide with those used in the fitting algorithm. Currently `exact=TRUE` is not implemented, but prints an error message telling the user how to achieve the exact predictions. This is done my rerunning the algorithm with the desired values interspersed (in order) with the values used in the original fit. This is easily achieved via the R command `lamba=sort(c(object$lambda, new.lambda))`

`...` Not used. Other arguments to predict.

Details

The shape of the objects returned are different for `"multinomial"` objects. This function actually calls `NextMethod()`, and the appropriate predict method is invoked for each of the three model types. `coef(...)` is equivalent to `predict(type="coefficients",...)`

Value

The object returned depends on type.

Author(s)

Jerome Friedman, Trevor Hastie and Rob Tibshirani
Maintainer: Trevor Hastie <hastie@stanford.edu>

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) *Regularization Paths for Generalized Linear Models via Coordinate Descent*

See Also

`glmnet`, and `print`, and `coef` methods.

Examples

```
x=matrix(rnorm(100*20),100,20)
y=rnorm(100)
g2=sample(1:2,100,replace=TRUE)
g4=sample(1:4,100,replace=TRUE)
fit1=glmnet(x,y)
predict(fit1,newx=x[1:5,],s=c(0.01,0.005))
predict(fit1,type="coef")
fit2=glmnet(x,g2,family="binomial")
predict(fit2,type="response",newx=x[2:5,])
predict(fit2,type="nonzero")
fit3=glmnet(x,g4,family="multinomial")
predict(fit3,newx=x[1:3,],type="response",s=0.01)
```

print.glmnet *print a glmnet object*

Description

Print a summary of the glmnet path at each step along the path.

Usage

```
## S3 method for class 'glmnet':
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

x	fitted glmnet object
digits	significant digits in printout
...	additional print arguments

Details

The call that produced the object x is printed, followed by a three-column matrix with columns Df, %dev and Lambda. The Df column is the number of nonzero coefficients (Df is a reasonable name only for lasso fits). %dev is the percent deviance explained (relative to the null deviance).

Value

The matrix above is silently returned

Author(s)

Jerome Friedman, Trevor Hastie and Rob Tibshirani
Maintainer: Trevor Hastie <hastie@stanford.edu>

References

Friedman, J., Hastie, T. and Tibshirani, R. (2008) *Regularization Paths for Generalized Linear Models via Coordinate Descent*

See Also

glmnet, predict and coef methods.

Examples

```
x=matrix(rnorm(100*20),100,20)
y=rnorm(100)
fit1=glmnet(x,y)
print(fit1)
```

Index