Package ‘HDtweedie’

February 19, 2015

Title
The Lasso for the Tweedie's Compound Poisson Model Using an IRLS-BMD Algorithm

Version 1.1
Date 2013-9-25
Author
Wei Qian <weiqian@umn.edu>, Yi Yang <yiyang@umn.edu>, Hui Zou <hzou@umn.edu>
Maintainer Wei Qian <weiqian@umn.edu>

Description
This package implements a iteratively reweighed least square (IRLS) strategy that incorporates a blockwise majorization decent (BMD) method, for efficiently computing the solution paths of the (grouped) lasso and the (grouped) elastic net for the Tweedie model.

License GPL-2
NeedsCompilation yes
Repository CRAN
Date/Publication 2013-10-22 08:15:32

R topics documented:

auto ................................................................. 2
coef.cv.HDtweedie ............................................ 3
cpy.HDtweedie .................................................. 5
cv.HDtweedie ................................................... 6
HDtweedie ....................................................... 8
plot.cv.HDtweedie ............................................. 11
plot.HDtweedie ............................................... 12
predict.cv.HDtweedie ......................................... 14
predict.HDtweedie ............................................ 15
print.HDtweedie ............................................... 17

Index 19
Description

The motor insurance dataset is originally retrieved from the SAS Enterprise Miner database. The included dataset is generated by re-organization and transformation as described in Qian et al. (2013).

Usage

data(auto)

Details

This data set contains 2812 policy samples with 56 predictors. See Qian et al. (2013) for a detailed description of the generation of these predictors. The response is the aggregate claim loss (in thousand dollars). The predictors are expanded from the following original variables:

- CAR_TYPE: car type, 6 categories
- JOBCLASS: job class, 8 categories
- MAX_EDUC: education level, 5 categories
- KIDSDRV: number of children passengers
- TRAVTIME: time to travel from home to work
- BLUEBOOK: car value
- NPOLICY: number of policies
- MVR_PTS: motor vehicle record point
- AGE: driver age
- HOMEKIDS: number of children at home
- YOJ: years on job
- INCOME: income
- HOME_VAL: home value
- SAMEHOME: years in current address
- CAR_USE: whether the car is for commercial use
- RED_CAR: whether the car color is red
- REVLKED: whether the driver’s license was revoked in the past
- GENDER: gender
- MARRIED: whether married
- PARENT1: whether a single parent
- AREA: whether the driver lives in urban area
Value

A list with the following elements:

- x: a [2812 x 56] matrix giving 2812 policy records with 56 predictors
- y: the aggregate claim loss

References


Examples

```r
# load HDTweedie library
library(HDTweedie)

# load data set
data(auto)

# how many samples and how many predictors?
dim(auto$x)

# response
auto$y
```

```r
coef.cv.HDtweedie  get coefficients or make coefficient predictions from a "cv.HDtweedie" object.
```

Description

This function gets coefficients or makes coefficient predictions from a cross-validated HDTweedie model, using the "cv.HDtweedie" object, and the optimal value chosen for lambda.

Usage

```r
# S3 method for class 'cv.HDtweedie'
coeff(object, s=c("lambda.1se","lambda.min"),...)
```
Arguments

- **object**: fitted `cv.HDtweedie` object.
- **s**: value(s) of the penalty parameter lambda at which predictions are required. Default is the value s="lambda.1se" stored on the CV object, it is the largest value of lambda such that error is within 1 standard error of the minimum. Alternatively s="lambda.min" can be used, it is the optimal value of lambda that gives minimum cross validation error cvm. If s is numeric, it is taken as the value(s) of lambda to be used.

Details

This function makes it easier to use the results of cross-validation to get coefficients or make coefficient predictions.

Value

The coefficients at the requested values for lambda.

Author(s)

Wei Qian, Yi Yang and Hui Zou

Maintainer: Wei Qian <weiqian@stat.umn.edu>

References


http://www.jstatsoft.org/v33/i01/

See Also

`cv.HDtweedie`, and `predict.cv.HDtweedie` methods.

Examples

```r
# load HDtweedie library
library(HDtweedie)

# load data set
data(auto)

# 5-fold cross validation using the lasso
cv6 <- cv.HDtweedie(x=auto$x,y=auto$y,p=1.5,nfolds=5)

# the coefficients at lambda = lambda.1se
```
```

calcus
# define group index
group1 <- c(rep(1,5),rep(2,7),rep(3,4),rep(4:14,each=3),15:21)

# 5-fold cross validation using the grouped lasso
cv1 <- cv.hdtweedie(x=auto$x,y=auto$y,group=group1,p=1.5,nfolds=5)

# the coefficients at lambda = lambda.min
calcus1, s = cv1$lambda.min

calcus(cus, s = NULL, ...)

Arguments

object fitted HDtweedie model object.
s value(s) of the penalty parameter lambda at which predictions are required. Default is the entire sequence used to create the model.
... not used. Other arguments to predict.

Details

s is the new vector at which predictions are requested. If s is not in the lambda sequence used for fitting the model, the coef function will use linear interpolation to make predictions. The new values are interpolated using a fraction of coefficients from both left and right lambda indices.

Value

The coefficients at the requested values for lambda.

Author(s)

Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>
```
References


See Also

`predict.HDtweedie` method

Examples

```r
# load HDtweedie library
library(HDtweedie)

# load data set
data(auto)

# fit the lasso
m0 <- HDtweedie(x=auto$x, y=auto$y, p=1.5)

# the coefficients at lambda = 0.01
coef(m0, s=0.01)

# define group index
group1 <- c(rep(1,5), rep(2,7), rep(3,4), rep(4:14, each=3), 15:21)

# fit grouped lasso
m1 <- HDtweedie(x=auto$x, y=auto$y, group=group1, p=1.5)

# the coefficients at lambda = 0.01 and 0.04
coef(m1, s=c(0.01, 0.04))
```

---

### cv.HDtweedie

**Cross-validation for HDtweedie**

**Description**

Does k-fold cross-validation for HDtweedie, produces a plot, and returns a value for lambda. This function is modified based on the cv function from the glmnet package.

**Usage**

```r
cv.HDtweedie(x, y, group = NULL, p, weights, lambda = NULL, pred.loss = c("deviance", "mae", "mse"), nfolds = 5, foldid, ...)
```
Arguments

- **x**: matrix of predictors, of dimension \( n \times p \); each row is an observation vector.
- **y**: response variable. This argument should be non-negative.
- **group**: To apply the grouped lasso, it is a vector of consecutive integers describing the grouping of the coefficients (see example below). To apply the lasso, the user can ignore this argument, and the vector is automatically generated by treating each variable as a group.
- **p**: the power used for variance-mean relation of Tweedie model. Default is 1.50.
- **weights**: the observation weights. Default is equal weight.
- **lambda**: optional user-supplied lambda sequence; default is NULL, and HDTweedie chooses its own sequence.
- **pred.loss**: loss to use for cross-validation error. Valid options are:
  - "deviance" Deviance.
  - "mae" Mean absolute error.
  - "mse" Mean square error.
  Default is "deviance".
- **nfolds**: number of folds - default is 5. Although nfolds can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets. Smallest value allowable is nfolds=3.
- **foldid**: an optional vector of values between 1 and nfolds identifying what fold each observation is in. If supplied, nfolds can be missing.
- ... other arguments that can be passed to HDTweedie.

Details

The function runs HDTweedie nfolds+1 times; the first to get the lambda sequence, and then the remainder to compute the fit with each of the folds omitted. The average error and standard deviation over the folds are computed.

Value

An object of class cv.HDTweedie is returned, which is a list with the ingredients of the cross-validation fit.

- **lambda**: the values of lambda used in the fits.
- **cvm**: the mean cross-validated error - a vector of length length(lambda).
- **cvsd**: estimate of standard error of cvm.
- **cvupper**: upper curve = cvm+cvsd.
- **cvlower**: lower curve = cvm-cvsd.
- **name**: a text string indicating type of measure (for plotting purposes).
- **HDTweedie.fit**: a fitted HDTweedie object for the full data.
- **lambda.min**: The optimal value of lambda that gives minimum cross validation error cvm.
- **lambda.1se**: The largest value of lambda such that error is within 1 standard error of the minimum.
Author(s)

Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>

References


See Also

`hdtweedie`, `plotNcvNhdtweedie`, `predictNcvNhdtweedie`, and `coefNcvNhdtweedie` methods.

Examples

```r
# load hdtweedie library
library(HDtweedie)

# load data set
data(auto)

# 5-fold cross validation using the lasso
cv0 <- cv.HDtweedie(x=auto$x,y=auto$y,p=1.5,nfolds=5)

# define group index
group1 <- c(rep(1,5),rep(2,7),rep(3,4),rep(4:14,each=3),15:21)

# 5-fold cross validation using the grouped lasso
cv1 <- cv.HDtweedie(x=auto$x,y=auto$y,group=group1,p=1.5,nfolds=5)
```

Fits the regularization paths for lasso-type methods of the Tweedie model

**Description**

Fits regularization paths for lasso-type methods of the Tweedie model at a sequence of regularization parameters lambda.

**Usage**

```r
HDtweedie(x, y, group = NULL,
p = 1.50,
weights = rep(1,nobs),
alpha = 1,
nlambda = 100,
lambda.factor = ifelse(nobs < nvars, 0.05, 0.001),
```
Arguments

x  
matrix of predictors, of dimension $n \times p$; each row is an observation vector.

y  
response variable. This argument should be non-negative.

group  
To apply the grouped lasso, it is a vector of consecutive integers describing the grouping of the coefficients (see example below). To apply the lasso, the user can ignore this argument, and the vector is automatically generated by treating each variable as a group.

p  
the power used for variance-mean relation of Tweedie model. Default is 1.50.

weights  
the observation weights. Default is equal weight.

alpha  
The elasticnet mixing parameter, with $0 \leq \alpha \leq 1$. The penalty is defined as

$$\frac{(1 - \alpha)}{2} ||\beta||_2^2 + \alpha ||\beta||_1.$$

alpha=1 is the lasso penalty, and alpha=0 the ridge penalty. Default is 1.

nlambda  
the number of lambda values - default is 100.

lambda.factor  
the factor for getting the minimal lambda in lambda sequence, where $\min(\lambda) = \lambda.factor \times \max(\lambda)$. $\max(\lambda)$ is the smallest value of lambda for which all coefficients are zero. The default depends on the relationship between $n$ (the number of rows in the matrix of predictors) and $p$ (the number of predictors). If $n \geq p$, the default is $0.001$, close to zero. If $n < p$, the default is $0.05$. A very small value of lambda.factor will lead to a saturated fit. It takes no effect if there is user-defined lambda sequence.

lambda  
a user supplied lambda sequence. Typically, by leaving this option unspecified users can have the program compute its own lambda sequence based on nlambda and lambda.factor. Supplying a value of lambda overrides this. It is better to supply a decreasing sequence of lambda values than a single (small) value. If not, the program will sort user-defined lambda sequence in decreasing order automatically.

pf  
penalty factor, a vector in length of bn (bn is the total number of groups). Separate penalty weights can be applied to each group to allow differential shrinkage. Can be 0 for some groups, which implies no shrinkage, and results in that group always being included in the model. Default value for each entry is the square-root of the corresponding size of each group (for the lasso, it is 1 for each variable).

dfmax  
limit the maximum number of groups in the model. Default is bs+1.

pmax  
limit the maximum number of groups ever to be nonzero. For example once a group enters the model, no matter how many times it exits or re-enters model through the path, it will be counted only once. Default is $\min(\text{dfmax} + 1.2, \text{bs})$. 

lambda = NULL,
pf = sqrt(bs),
dfmax = as.integer(max(group)) + 1,
pmax = min(dfmax * 1.2, as.integer(max(group))),
standardize = FALSE,
eps = 1e-08, maxit = 3e+08)
eps convergence termination tolerance. Defaults value is 1e-8.

standardize logical flag for variable standardization, prior to fitting the model sequence. If TRUE, x matrix is normalized such that each column is centered and sum squares of each column \( \sum_{i=1}^{N} x_{ij}^2 / N = 1 \). The coefficients are always returned on the original scale. Default is FALSE.

maxit maximum number of inner-layer BMD iterations allowed. Default is 3e8.

Details

The sequence of models implied by lambda is fit by the IRLS-BMD algorithm. This gives a (grouped) lasso or (grouped) elasticnet regularization path for fitting the Tweedie generalized linear regression paths, by maximizing the corresponding penalized Tweedie log-likelihood. If the group argument is ignored, the function fits the lasso. Users can tweak the penalty by choosing different alpha and penalty factor.

For computing speed reason, if models are not converging or running slow, consider increasing eps, decreasing nlambda, or increasing lambda.factor before increasing maxit.

Value

An object with S3 class HDtweedie.

call the call that produced this object

b0 intercept sequence of length length(lambda)

beta a p*length(lambda) matrix of coefficients.

df the number of nonzero groups for each value of lambda.

dim dimension of coefficient matrix (ices)

lambda the actual sequence of lambda values used

npasses total number of iterations (the most inner loop) summed over all lambda values

jerr error flag, for warnings and errors, 0 if no error.

group a vector of consecutive integers describing the grouping of the coefficients.

Author(s)

Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>

References


See Also

plot.HDtweedie
Examples

# load HDtweedie library
library(HDtweedie)

# load auto data set
data(auto)

# fit the lasso
m0 <- HDtweedie(x=auto$x, y=auto$y, p=1.5)

# define group index
group1 <- c(rep(1,5), rep(2,7), rep(3,4), rep(4:14, each=3), 15:21)

# fit the grouped lasso
m1 <- HDtweedie(x=auto$x, y=auto$y, group=group1, p=1.5)

# fit the grouped elastic net
m2 <- HDtweedie(x=auto$x, y=auto$y, group=group1, p=1.5, alpha=0.7)

plot.cv.HDtweedie  

plot the cross-validation curve produced by cv.HDtweedie

Description

Plots the cross-validation curve, and upper and lower standard deviation curves, as a function of the lambda values used. This function is modified based on the plot.cv function from the glmnet package.

Usage

## S3 method for class 'cv.HDtweedie'
plot(x, sign.lambda, ...)

Arguments

x  
fitted cv.HDtweedie object

sign.lambda  
either plot against log(lambda) (default) or its negative if sign.lambda=-1.

...  
other graphical parameters to plot

Details

A plot is produced.

Author(s)

Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>
plot.HDtweedie

Plot solution paths from a "HDtweedie" object

Description

Produces a coefficient profile plot of the coefficient paths for a fitted HDtweedie object.

Usage

## S3 method for class 'HDtweedie'
plot(x, group = FALSE, log.l = TRUE, ...)

References


See Also
cv.HDtweedie.
Arguments

- **x**: fitted `HDtweedie` model
- **group**: what is on the Y-axis. Plot the norm of each group if `TRUE`. Plot each coefficient if `FALSE`.
- **log.l**: what is on the X-axis. Plot against the log-lambda sequence if `TRUE`. Plot against the lambda sequence if `FALSE`.
- ... other graphical parameters to plot

Details

A coefficient profile plot is produced.

Author(s)

Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>

References


Examples

```r
# load HDtweedie library
library(HDtweedie)

# load data set
data(auto)

# fit the lasso
m0 <- HDtweedie(x=auto$x, y=auto$y, p=1.5)

# make plot
plot(m0) # plots the coefficients against the log-lambda sequence

# define group index
group1 <- c(rep(1,5), rep(2,7), rep(3,4), rep(4:14, each=3), 15:21)

# fit group lasso
m1 <- HDtweedie(x=auto$x, y=auto$y, group=group1, p=1.5)

# make plots
par(mfrow=c(1,3))
plot(m1) # plots the coefficients against the log-lambda sequence
plot(m1, group=TRUE) # plots group norm against the log-lambda sequence
plot(m1, log.l=FALSE) # plots against the lambda sequence
```
predict.cv.HDtweedie

make predictions from a "cv.HDtweedie" object.

Description
This function makes predictions from a cross-validated HDtweedie model, using the stored "cv.HDtweedie" object, and the optimal value chosen for lambda.

Usage

## S3 method for class 'cv.HDtweedie'
predict(object, newx, s=c("lambda.1se","lambda.min"),...)

Arguments

- **object**: fitted `cv.HDtweedie` object.
- **newx**: matrix of new values for x at which predictions are to be made. Must be a matrix. See documentation for `predict.HDtweedie`.
- **s**: value(s) of the penalty parameter lambda at which predictions are required. Default is the value s="lambda.1se" stored on the CV object. Alternatively s="lambda.min" can be used. If s is numeric, it is taken as the value(s) of lambda to be used.
- ... not used. Other arguments to predict.

Details
This function makes it easier to use the results of cross-validation to make a prediction.

Value
The returned object depends on the ...argument which is passed on to the `predict` method for `HDtweedie` objects.

Author(s)
Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>

References

See Also
`cv.HDtweedie`, and `coef.cv.HDtweedie` methods.
Examples

```r
# load HDtweedie library
library(HDtweedie)

# load data set
data(auto)

# 5-fold cross validation using the lasso
cv0 <- cv.HDtweedie(x=auto$x,y=auto$y,p=1.5,nfolds=5)

# predicted mean response at lambda = lambda.1se, newx = x[1,]
pre = predict(cv0, newx = auto$x[1,], type = "response")

# define group index
group1 <- c(rep(1,5),rep(2,7),rep(3,4),rep(4:14,each=3),15:21)

# 5-fold cross validation using the grouped lasso
cv1 <- cv.HDtweedie(x=auto$x,y=auto$y,group=group1,p=1.5,nfolds=5)

# predicted the log mean response at lambda = lambda.min, x[1:5,]
pre = predict(cv1, newx = auto$x[1:5,], s = cv1$lambda.min, type = "link")
```

predict.HDtweedie  

**make predictions from a "HDtweedie" object.**

Description

Similar to other predict methods, this function predicts fitted values from a HDtweedie object.

Usage

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

Arguments

- `object` fitted HDtweedie model object.
- `newx` matrix of new values for x at which predictions are made. Must be a matrix.
- `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 "response" gives the mean response estimate.
  - Type "link" gives the estimate for log mean response.
- `...` Not used. Other arguments to predict.
Details

s is the new vector at which predictions are requested. If s is not in the lambda sequence used for fitting the model, the predict function will use linear interpolation to make predictions. The new values are interpolated using a fraction of predicted values from both left and right lambda indices.

Value

The object returned depends on type.

Author(s)

Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>

References


See Also

coef method

Examples

# load HDtweedie library
library(HDtweedie)

# load auto data set
data(auto)

# fit the lasso
m0 <- HDtweedie(x=auto$x,y=auto$y,p=1.5)

# predicted mean response at x[10,]
print(predict(m0,type="response",newx=auto$x[10,]))

# define group index
group1 <- c(rep(1,5),rep(2,7),rep(3,4),rep(4:14,each=3),15:21)

# fit the grouped lasso
m1 <- HDtweedie(x=auto$x,y=auto$y,group=group1,p=1.5)

# predicted the log mean response at x[1:5,]
print(predict(m1,type="link",newx=auto$x[1:5,]))
print.HDtweedie

Description

Print the nonzero group counts at each lambda along the HDtweedie path.

Usage

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

Arguments

- `x`: fitted HDtweedie object
- `digits`: significant digits in printout
- `...`: additional print arguments

Details

Print the information about the nonzero group counts at each lambda step in the HDtweedie object. The result is a two-column matrix with columns `df` and `lambda`. The `df` column is the number of the groups that have nonzero within-group coefficients, the `lambda` column is the the corresponding lambda.

Value

a two-column matrix, the first columns is the number of nonzero group counts and the second column is `lambda`.

Author(s)

Wei Qian, Yi Yang and Hui Zou
Maintainer: Wei Qian <weiqian@stat.umn.edu>

References

Examples

```r
# load HDtweedie library
library(HDtweedie)

# load auto data set
data(auto)

# fit the lasso
m0 <- HDtweedie(x=auto$x, y=auto$y, p=1.5)

# print out results
print(m0)

# define group index
group1 <- c(rep(1,5),rep(2,7),rep(3,4),rep(4:14,each=3),15:21)

# fit the grouped lasso
m1 <- HDtweedie(x=auto$x, y=auto$y, group=group1, p=1.5)

# print out results
print(m1)
```
Index

+Topic **datasets**
  auto, 2

+Topic **models**
  coef.cv.HDtweedie, 3
  coef.HDtweedie, 5
  cv.HDtweedie, 6
  HDtweedie, 8
  plot.cv.HDtweedie, 11
  plot.HDtweedie, 12
  predict.cv.HDtweedie, 14
  predict.HDtweedie, 15
  print.HDtweedie, 17

+Topic **regression**
  coef.cv.HDtweedie, 3
  coef.HDtweedie, 5
  cv.HDtweedie, 6
  HDtweedie, 8
  plot.cv.HDtweedie, 11
  plot.HDtweedie, 12
  predict.cv.HDtweedie, 14
  predict.HDtweedie, 15
  print.HDtweedie, 17

auto, 2

coef, 16
coef.cv.HDtweedie, 3, 8, 14
coef.HDtweedie, 5
cv.HDtweedie, 4, 6, 7, 11, 12, 14

HDtweedie, 5, 7, 8, 8, 10, 12–15, 17

plot.cv.HDtweedie, 8, 11
plot.HDtweedie, 12
predict, 14
predict.cv.HDtweedie, 4, 8, 14
predict.HDtweedie, 6, 15
print.HDtweedie, 17