Package ‘msda’

February 20, 2015

Title  Multi-Class Sparse Discriminant Analysis
Version  1.0.2
Date  2014-09-26
Author  
Qing Mai <mai@stat.fsu.edu>, Yi Yang <yiyang@umn.edu>, Hui Zou <hzou@stat.umn.edu>
Maintainer  Yi Yang <yiyang@umn.edu>
Depends  Matrix, MASS
Description  Efficient procedures for computing a new Multi-Class Sparse Discriminant Analysis method that estimates all discriminant directions simultaneously.
LazyData  yes
License  GPL-2
URL  https://github.com/emeryyi/msda
NeedsCompilation  yes
Repository  CRAN
Date/Publication  2015-02-20 09:16:53

R topics documented:
cv.msda  ................................................................. 2
GDS1615  ............................................................... 3
msda  ................................................................. 4
plot.msda  ............................................................ 6
predict.msda  ......................................................... 7

Index  9
cv.msda  Cross-validation for msda

Description

Does k-fold cross-validation for msda, returns a value for lambda.

Usage

```
cv.msda(x, y, nfolds = 5, lambda = NULL, lambda.opt = "min", ...)
```

Arguments

- `x`: matrix of predictors, of dimension $N \times p$; each row is an observation vector.
- `y`: response variable. This argument should be a factor for classification.
- `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`.
- `lambda`: optional user-supplied lambda sequence; default is `NULL`, and `msda` chooses its own sequence.
- `lambda.opt`: If choose "min", the smallest lambda that gives minimum cross validation error `cvm` will be returned. If choose "max", the largest lambda that gives minimum cross validation error `cvm` will be returned.
- `...`: other arguments that can be passed to msda.

Details

The function runs `msda` `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.msda` 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`.
- `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.
- `msda.fit`: a fitted `msda` object for the full data.
Author(s)
Qing Mai <mai@stat.fsu.edu>, Yi Yang <yiyang@umn.edu>, Hui Zou <hzhou@stat.umn.edu>
Maintainer: Yi Yang <yiyang@umn.edu>

References
URL: https://github.com/emeryyi/msda

See Also
msda

Examples
data(GDS1615)
x<-GDS1615$x
y<-GDS1615$y
obj.cv<-cv.msda(x=x,y=y,nfolds=5,lambda.opt="max")
lambda.min<-obj.cv$lambda.min
id.min<-which(obj.cv$lambda==lambda.min)
pred<-predict(obj.cv$msda.fit,x)[,id.min]

GDS1615
GDS1615 data introduced in Burczynski et al. (2012).

Description
The dataset is a subset of the dataset available on Gene Expression Omnibus with the accession number GDS1615. The original dataset contains 22283 gene expression levels and the disease states of the observed subjects. In Mai, Yang and Zou, the dimension of the original dataset was first reduced to 127 by F-test screening.

Usage
data(GDS1615)

Value
This data frame contains the following:
x gene expression levels.
y Disease state that is coded as 1,2,3. 1: normal; 2: ulcerative colitis; 3: Crohn’s disease.
References


URL: https://github.com/emeryyi/msda

Examples

data(GDS1615)

msda(x, y, nlambda = 100,
 lambda.factor = ifelse((nobs - nclass) <= nvars, 0.2, 0.001),
 lambda = NULL, dfmax = nobs, pmax = min(dfmax * 2 + 20, nvars),
 pf = rep(1, nvars), eps = 1e-04, maxit = 1e+06, sml = 1e-06,
 verbose = FALSE, perturb = NULL)

Arguments

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

y
response variable. This argument should be a factor for classification.

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 $N > p$, the default is 0.0001, close to zero. If $N < p$, the default is 0.2. 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.

dfmax

limit the maximum number of variables in the model. Useful for very large p, if a partial path is desired. Default is n.

pmax

limit the maximum number of variables ever to be nonzero. For example once \( \beta \) 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 \( \min(dfmax*1.2,p) \).

pf

L1 penalty factor of length p. Separate L1 penalty weights can be applied to each coefficient of \( \theta \) to allow differential L1 shrinkage. Can be 0 for some variables, which implies no L1 shrinkage, and results in that variable always being included in the model. Default is 1 for all variables (and implicitly infinity for variables listed in exclude).

eps

convergence threshold for coordinate descent. Each inner coordinate descent loop continues until the relative change in any coefficient. Defaults value is 1e-8.

maxit

maximum number of outer-loop iterations allowed at fixed lambda value. Default is 1e6. If models do not converge, consider increasing maxit.

sml

verbose

whether to print out computation progress. The default is FALSE.

perturb

a scalar number. If it is specified, the number will be added to each diagonal element of the sigma matrix as perturbation. The default is NULL.

Details

Note that for computing speed reason, if models are not converging or running slow, consider increasing eps and sml, or decreasing nlambda, or increasing lambda.factor before increasing maxit. Users can also reduce dfmax to limit the maximum number of variables in the model.

Value

An object with S3 class msda.

theta

a list of length(nlambda) for fitted coefficients theta, each one corresponding to one lambda value, each stored as a sparse matrix (dgCMatrix class, the standard class for sparse numeric matrices in the Matrix package.). To convert it into normal type matrix use as.matrix().

df

the number of nonzero coefficients for each value of lambda.

obj

the fitted value of the objective function for each value of lambda.

dim

dimension of each coefficient matrix at each lambda.

lambda

the actual sequence of lambda values used.

x

matrix of predictors.
### Example

```r
data(GDS1615)
x <- GDS1615$x
y <- GDS1615$y
obj <- msda(x = x, y = y)
```
predict.msda

Arguments

- `x` : fitted `msda` model
- `xvar` : the variable on the X-axis. The option "norm" plots the coefficients against the L1-norm of the coefficients, and the option "lambda" plots the coefficient against the log-lambda sequence.
- ... : other graphical parameters to plot

Details

A coefficient profile plot is produced.

Author(s)

Qing Mai <mai@stat.fsu.edu>, Yi Yang <yiyang@umn.edu>, Hui Zou <hzou@stat.umn.edu>
Maintainer: Yi Yang <yiyang@umn.edu>

References


URL: [https://github.com/emeryyi/msda](https://github.com/emeryyi/msda)

Examples

data(GDS1615)
x<-GDS1615$x
y<-GDS1615$y
obj <- msda(x = x, y = y)
plot(obj)

```
predict.msda
make predictions from a "msda" object.
```

Description

This functions predicts class labels from a fitted `msda` object.

Usage

```
## S3 method for class 'msda'
predict(object, newx, ...)
```
Arguments

- **object**: fitted `msda` model object.
- **newx**: matrix of new values for `x` at which predictions are to be made. NOTE: `newx` must be a matrix, `predict` function does not accept a vector or other formats of `newx`.
- **...**: Not used. Other arguments to predict.

Value

Predicted class label(s) at the entire sequence of the penalty parameter `lambda` used to create the model.

Author(s)

Qing Mai <mai@stat.fsu.edu>, Yi Yang <yiyang@umn.edu>, Hui Zou <hzou@stat.umn.edu>

Maintainer: Yi Yang <yiyang@umn.edu>

References


URL: [https://github.com/emeryyi/msda](https://github.com/emeryyi/msda)

See Also

- `msda`

Examples

```r
data(GDS1615)
x <- GDS1615$x
y <- GDS1615$y
obj <- msda(x = x, y = y)
pred <- predict(obj, x)
```
Index

*Topic **classification**
  cv.msda, 2
  msda, 4
  plot.msda, 6
  predict.msda, 7
*Topic **datasets**
  GDS1615, 3
*Topic **models**
  cv.msda, 2
  msda, 4
  plot.msda, 6
  predict.msda, 7
  cv.msda, 2, 2
  GDS1615, 3
  msda, 2, 3, 4, 5–8
  plot.msda, 6
  predict.msda, 7
  x (GDS1615), 3
  y (GDS1615), 3