Package 'VDA'

February 19, 2015

Type Package

Title VDA

Version 1.3

Date 2013-07-05

Author Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer Edward Grant <edward.m.grant@gmail.com>

Description Multicategory Vertex Discriminant Analysis: A novel supervised multicategory classification method

License GPL-2

LazyLoad yes

LazyData yes

Depends rgl

NeedsCompilation yes

Repository CRAN

Date/Publication 2013-07-09 08:07:32

R topics documented:

`cv.vda.le` *Choose the optimal pair of lambdas, λ_1 and λ_2*

Description

Use k-fold validation to choose the optmial values for the tuning parameters λ_1 and λ_2 to be used in Multicategory Vertex Discriminant Analysis (`vda.le`).

Usage

```
cv.vda.le(x, y, kfold, lam.vec.1, lam.vec.2)
```

Arguments

`x`	*n* x *p* matrix or data frame containing the cases for each feature. The rows correspond to cases and the columns to the features. Intercept column is not included in this.
`y`	*n* x 1 vector representing the outcome variable. Each element denotes which one of the *k* classes that case belongs to.
`kfold`	The number of folds to use for the k-fold validation for each set of λ_1 and λ_2
`lam.vec.1`	A vector containing the set of all values of λ_1, from which VDA will be conducted. To use only Euclidean penalization, set `lam.vec.2`=0.
`lam.vec.2`	A vector containing the set of all values of λ_2, from which VDA will be conducted. vda.le is relatively insensitive to lambda values, so it is recommended that a vector of few values is used. The default value is 0.01. To use only Lasso penalization, set `lam.vec.1`=0.

Details

For each pair of (λ_1, λ_2), k-fold cross-validation will be conducted and the corresponding average testing error over the k folds will be recorded. λ_1 represents the parameter for the lasso penalization, while λ_2 represents the parameter for the group euclidean penalization. To use only Lasso penalization, set `lam.vec.2`=0. To use only Euclidean penalization, set `lam.vec.1`=0. The optimal pair is considered the pair of values that give the smallest testing error over the cross validation.

To view a plot of the cross validation errors across lambda values, see `plot.cv.vda.le`.

Value

`kfold`	The number of folds used in k-fold cross validation
`lam.vec.1`	The user supplied vector of λ_1 values
`lam.vec.2`	The user supplied vector of λ_2 values
`error.cv`	A matrix of average testing errors. The rows correspond to λ_1 values and the columns correspond to λ_2 values.
`lam.opt`	The pair of λ_1 and λ_2 values that return the lowest testing error across k-fold cross validation.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Wu, T.T. and Lange, K. (2010) Multicategory Vertex Discriminant Analysis for High-Dimensional Data. Annals of Applied Statistics, Volume 4, No 4, 1698-1721.

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

vda.le.

plot.cv.vda.le.

Examples

```
### load zoo data
### column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

### feature matrix
x <- zoo[,2:17]

### class vector
y <- zoo[,18]

### lambda vector
lam1 <- (1:5)/100
lam2 <- (1:5)/100

### Searching for the best pair, using both lasso and euclidean penalizations
cv <- cv.vda.le(x, y, kfold = 3, lam.vec.1 = exp(1:5)/10000, lam.vec.2 = (1:5)/100)
plot(cv)
outLE <- vda.le(x,y,cv$lam.opt[1],cv$lam.opt[2])

### To search for the best pair, using ONLY lasso penalization, set lambda2=0 (remove comments)
#cvlasso <- cv.vda.le(x, y, kfold = 3, lam.vec.1 = exp(1:10)/1000, lam.vec.2 = 0)
#plot(cvlasso)
#cvlasso$lam.opt

### To search for the best pair, using ONLY euclidean penalization, set lambda1=0 (remove comments)
#cveuclidian <- cv.vda.le(x, y, kfold = 3, lam.vec.1 = 0, lam.vec.2 = exp(1:10)/1000)
#plot(cveuclidian)
#cveuclidian$lam.opt

### Predict five cases based on vda.le (Lasso and Euclidean penalties)
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
```

```
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(outLE, fivecases)
```

`cv.vda.r` *Choose λ using K-fold cross validation*

Description

Choose the optimal tuning parameter λ for Vertex Discriminant Analyis by using K-fold cross validation.

Usage

```
cv.vda.r(x, y, k, lam.vec)
cv.vda(x, y, k, lam.vec)
```

Arguments

`x`	n x p matrix or data frame containing the cases for each feature. The rows correspond to cases and the columns to the features. Intercept column is not included in this.
`y`	n x 1 vector representing the outcome variable. Each element denotes which one of the k classes that case belongs to.
`k`	The number of folds to be used in cross-validation.
`lam.vec`	A vector containing the set of all values of λ, from which VDA will be conducted.

Details

K-fold cross validation to select optimal `lambda` for use in Vertex Disciminant Analysis (`vda.r`). The optimal value is considered the `lamda` value that retuns the lowest testing error over the cross validation. If more than one `lambda` value give the minumum testing error, the largest `lambda` is selected.

A plot of the cross validation errors can be viewed through `plot.cv.vda.r`.

Value

`k`	The value of `K` used for the K-fold cross validation.
`lam.vec`	The values of lambda tested.
`mean.error`	The mean error corresponding to each lambda across k-folds
`lam.opt`	The determined lambda value among `lam.vec` that returns the smallest prediction error. This value is the optimal `lambda` value for use in `link{vda.r}`.
`error.cv`	The prediction error matrix returned by cross validation method.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

vda.r. plot.cv.vda.r

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

# feature matrix without intercept
x <- zoo[,2:17]

# class vector
y <- zoo[,18]

# lambda vector
lam.vec <- (1:10)/10

# searching for the best lambda with 10-fold cross validation and plot cv
cv <- cv.vda.r(x, y, 10, lam.vec)
plot(cv)

# run VDA
out <- vda.r(x,y,cv$lam.opt)

# Predict five cases based on VDA
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(out, fivecases)
```

---

plot.cv.vda.le *Plot a cv.vda.le object*

Description

Plot a the cross validation error across lambda values

Usage

```
## S3 method for class 'cv.vda.le'
plot(x, ...)
```

Arguments

x	Object of class 'cv.vda.le', the result of a call to cv.vda.le.
...	Not used.

Details

3D plots the k-fold cross validation testing error for values across a different lambda1 and lambda2 values. Use cv.vda.le to produce the object of class "cv.vda.le".

When lam.vec.1 or lam.vec.2 is set to 0, the a 2D plot will be produced.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

vda.le, cv.vda.le

Examples

```
### load zoo data
### column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

### feature matrix without intercept
x <- zoo[,2:17]

### class vector
y <- zoo[,18]

### lambda vector
lam1 <- (1:5)/100
lam2 <- (1:5)/100

### searching for the best pair, using both lasso and euclidean penalizations
cv <- cv.vda.le(x, y, kfold=3, lam.vec.1=lam1, lam.vec.2=lam2)
plot(cv)
outLE <- vda.le(x,y,cv$lam.opt[1],cv$lam.opt[2])
```

```
### searching for the best pair, using ONLY lasso penalization, set lambda 2=0 (remove comments)
#cvlasso <- cv.vda.le(x, y, kfold=3, lam.vec.1=exp(1:10)/1000, lam.vec.2=0)
#plot(cvlasso)
#cvlasso$lam.opt

### searching for the best pair, using ONLY euclidean penalization, set lambda1=0 (remove comments)
#cveuclidian <- cv.vda.le(x, y, kfold=3, lam.vec.1=0, lam.vec.2=exp(1:10)/1000)
#plot(cveuclidian)
#cveuclidian$lam.opt

# Predict five cases based on vda.le (Lasso and Euclidean penalties)
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(outLE, fivecases)
```

`plot.cv.vda.r` *Plot a cv.vda.r object*

Description

Plot a the cross validation error across lambda values

Usage

```
## S3 method for class 'cv.vda.r'
plot(x, ...)
```

Arguments

`x`	Object of class 'cv.vda.r', the result of a call to `cv.vda.r`.
`...`	Not used.

Details

Plots the k-fold cross validation testing error for values across a different lambda values. Use `cv.vda.r` to produce the object of class "cv.vda.r."

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

vda.r, cv.vda.r

Examples

```
# load data
data(zoo)

# feature matrix without intercept
x <- zoo[,2:17]

# class vector
y <- zoo[,18]

# lambda vector
lam.vec <- (1:10)/10

# run 10 fold cross validation across lambdas
cv <- cv.vda.r(x, y, 10, lam.vec)

# plot CV results
plot(cv)

# Perform VDA with CV-selected optimal lambda
out <- vda.r(x,y,cv$lam.opt)

# Predict five cases based on VDA
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(out, fivecases)
```

`predict.vda.le` *Predict a vda.le object.*

Description

The predict function for a vda.le object.

Usage

```
## S3 method for class 'vda.le'
predict(object, newdata=NULL, ...)
```

Arguments

object	An object of class 'vda.le', usually the result of a call to vda.le.
newdata	An optional n x p matrix or data frame containing new data to be classified using vertex discriminant analysis. The data must contain the same number of attributes as the training data. If newdata is omitted, the training data is used.
...	Not used.

Details

The prediction funtion for Vertex Discriminant Analysis (vda.le). Returns 1 x n vector in which each element represents the predicted value for the corresponding case.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

vda.le, summary.vda.le, print.vda.le

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

# feature matrix without intercept
x <- zoo[,2:17]

# class vector
y <- zoo[,18]

# run VDA
out <- vda.le(x,y)

# predict cases based on VDA
onecase <- matrix(c(0,0,1,0,0,1,1,0,0,0,0,0,6,0,0,0),nrow=1)

fivecases <- matrix(0,5,16)
```

```
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(out, fivecases)
```

`predict.vda.r` *Predict a vda.r object.*

Description

The predict function for a vda.r object.

Usage

```
## S3 method for class 'vda.r'
predict(object, newdata=NULL, ...)
```

Arguments

`object`	An object of class 'vda.r', usually the result of a call to `vda.r`.
`newdata`	An optional *n* x *p* matrix or data frame containing new data to be classified using VDA. The data must contain the same number of attributes as the training data. If `newdata` is omitted, the training data is used.
`...`	Not used.

Details

The prediction funtion for Vertex Discriminant Analysis (`vda.r`). Returns 1 x *n* vector in which each element represents the predicted value for the corresponding case.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

`vda.r`, `summary.vda.r`, `print.vda.r`

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

# feature matrix without intercept
x <- zoo[,2:17]

# class vector
y <- zoo[,18]

# run VDA
out <- vda.r(x,y)

# predict cases based on VDA
onecase <- matrix(c(0,0,1,0,0,1,1,0,0,0,0,0,6,0,0,0),nrow=1)

fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(out, fivecases)
```

`print.vda.le` *Print a vda.le object*

Description

The default print method for a vda.le object.

Usage

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

Arguments

`x`	Object of class 'vda.le', usually the result of a call to `vda.le`.
`...`	Not used.

Details

Prints out the predicted classes for given training data found using Vertex Discriminant Analysis. `summary.vda.le` provides more detailed information about the VDA object `x`.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

vda.le, summary.vda.le

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

# feature matrix without intercept
x <- zoo[,2:17]

# class vector
y <- zoo[,18]

#run VDA
out <- vda.le(x, y)

print(out)
```

---

print.vda.r *Print a vda.r object*

Description

The default print method for a vda.r object.

Usage

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

Arguments

x	Object of class 'vda.r', usually the result of a call to vda.r.
...	Not used.

Details

Prints out the predicted classes for given training data found using Vertex Discriminant Analysis. summary.vda.r provides more detailed information about the VDA object.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

vda.r, summary.vda.r

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

# feature matrix without intercept
x <- zoo[,2:17]

# class vector
y <- zoo[,18]

#run VDA
out <- vda.r(x, y)

print(out)
```

---

summary.vda.le *Summary for a vda.le object*

Description

Takes a fitted VDA object produced by vda.le and produces various useful summaries from it.

Usage

```
## S3 method for class 'vda.le'
summary(object, ...)
```

Arguments

object	An object of class 'vda.le', usually the result of a call to vda.le.
...	Not used.

Details

The function prints the number of cases, the number of classes, and the number of features in object, of class vda.le. It also prints the lambda used in the analysis. Additionally, it prints the coefficients and the resulting predictions made by Vertex Discriminant Analysis on the training set and the following training error.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

See Also

vda.le, print.vda.le

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

# feature matrix without intercept
x<-zoo[,2:17]

# class vector
y<-zoo[,18]

#run VDA
out<-vda.le(x, y)

summary(out)
```

---

summary.vda.r *Summary for a vda.r object*

Description

Takes a fitted vda.r object produced by vda.r and produces various useful summaries from it.

Usage

```
## S3 method for class 'vda.r'
summary(object, ...)
```

Arguments

object	An object of class 'vda.r', usually the result of a call to vda.r.
...	Not used.

Details

The function prints the number of cases, the number of classes, and the number of features in object, of class vda.r. It also prints the lambda used in the analysis. Additionally, it prints the coefficients and the resulting predictions made by Vertex Discriminant Analysis on the training set and the following training error.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

See Also

vda.r, print.vda.r

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

# feature matrix without intercept
x<-zoo[,2:17]

# class vector
y<-zoo[,18]

#run VDA
out<-vda.r(x, y)

summary(out)
```

VDA *Multicategory Vertex Discriminant Analysis*

Description

This package provides functions to optimize and execute Multicategory Vertex Discriminant Analysis, a method of supervised learning for an outcome with k predictor categories. Outcome classification is based on linear discrimination among the vertices of a regular simplex in a k-1-dimension Euclidean space, where each vertex represents a different category.

Details

Package:	VDA
Type:	Package
Version:	1.3
Date:	2013-Jul-05
License:	GPL-2
LazyLoad:	yes

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

Examples

```
#load dataset from package
data(zoo)

#matrix containing all predictor vectors
x <- zoo[,2:17]

#outcome class vector
y <- zoo[,18]

#run VDA (ridge penalty)
out <- vda.r(x, y)

#Predict five cases based on VDA
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(out, fivecases)

#run vda.le (lasso and euclidean penalty)
outLE <- vda.le(x, y)

#Predict five cases based on VDA
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
```

```
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(outLE, fivecases)
```

`vda.le` *Multicategory Vertex Discriminant Analysis (VDA) For High-Dimensional Data*

Description

The method of vertex discriminant analysis (VDA) is ideally suited to handle multiple categories and an excess of predictors over training cases. `vda.le` is an elaboration of VDA that simultaneously conducts classification of k possible categories and variable selection of p features, based on a data set of n cases. Variable selection is imposed using *L1* (Lasso) and group Euclidean penalties. To use only Lasso penalization, set $\lambda 2$=0. To use only Euclidean penalization, set $\lambda 1$=0.

Usage

```
vda.le(x, y, lambda1, lambda2)
```

Arguments

`x`	n x p matrix or data frame containing the cases for each feature. The rows correspond to cases and the columns to the features. Intercept column is not included in this.
`y`	n x 1 vector representing the outcome variable. Each element denotes which one of the k classes that case belongs to
`lambda1`	Tuning parameter to control the lasso penalty. The default value is $1/n$. For determining the optimal `lambda1`, refer to `cv.vda.le`.
`lambda2`	Tuning parameter to control the Euclidean penalty. The default value is 0.01. For determining the optimal `lambda1`, refer to `cv.vda.le`

Details

`vda.le` carries out cyclic coordinate descent in the context of VDA to minimize the loss function. By adding lasso ($L1$-norm) and group Euclidean penalties to the VDA loss function, unnecessary predictors are elimnated, adding parsimony and making the model more interpretable. Lasso penalties are applied to each predictor coefficient separately, while Euclidean penalties couples the coefficients of a single predictor and penalize the group collectively. If `lambda1=0`, then the overall penalty reduces to only group penalties. When `lambda2=0`, then the overall penalty reduces to the lasso. With these penalties in place, cyclic coordinate descent accelerates estimation of all coefficients.

Value

`feature`	Feature matrix `x` with an intercept vector added as the first column. All entries in the first column should equal 1.
`stand.feature`	The feature matrix where the all columns are standardized, with the exception of the intercept column which is left unstandardized.
`class`	Class vector `y`. All elements should be integers between 1 and `classes`.
`cases`	Number of cases, *n*.
`classes`	Number of classes, *k*.
`features`	Number of feautres, *p*.
`lambda`	Vector of tuning constants where the first component is `lambda1` and the second is `lambda2`
`predicted`	Vector of predicted category values based on VDA.
`coefficient`	The estimated coefficient matrix where the columns represent the coefficients for each predictor variable corresponding to `k-1` outcome categories. The coefficient matrix is used for classifying new cases.
`training_error_rate`	The percentage of instances in the training set where the predicted outcome category is not equal to the case's true category.
`nonzeros`	Number of feature coefficients retained in the model. Is equal to *p* - number of features eliminated by penalization.
`selected`	An integer vector which represents the attributes that were selected after penalization.
`call`	The matched call.

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Wu, T.T. and Lange, K. (2010) Multicategory Vertex Discriminant Analysis for High-Dimensional Data. Annals of Applied Statistics, Volume 4, No 4, 1698-1721.

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

For determining the optimal values for `lambda1` and `lambda2`, see `cv.vda.le`

For VDA without variable selection, see `vda.r`.

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

#matrix containing all predictor vectors
x <- zoo[,2:17]

#outcome class vector
y <- zoo[,18]

#run VDA, Only Lasso Penalization, Set lambda2=0
outlasso <-vda.le(x,y,lambda1=.02,lambda2=0)

#run VDA, Only Euclidean Penalization, Set lambda1=0
outeuclid <-vda.le(x,y,lambda1=0,lambda2=0.04)

#run VDA, Lasso and Euclidean Penalization
outLE<-vda.le(x,y,lambda1=0.009,lambda2=0.05)
summary(outLE)

#Predict five cases based on VDA, Lasso and Euclidean Penalization
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(outLE, fivecases)
```

---

vda.r *Vertex Discriminant Analysis*

Description

Multicategory Vertex Discriminant Analysis (VDA) for classifying an outcome with k possible categories and p features, based on a data set of n cases. The default penalty function is Ridge. Lasso, Euclidean, and a mixture of Lasso and Euclidean are also available. Please refer to vda.le

Usage

```
vda.r(x, y, lambda)
vda(x, y, lambda)
```

Arguments

x	*n* x *p* matrix or data frame containing the cases for each feature. The rows correspond to cases and the columns to the features. Intercept column is not included in this.

y	*n* x 1 vector representing the outcome variable. Each element denotes which one of the *k* classes that case belongs to
lambda	Tuning constant. The default value is set as $1/n$. Can also be found using `cv.vda.r`, which uses K-fold cross validation to determine the optimal value.

Details

Outcome classification is based on linear discrimination among the vertices of a regular simplex in a *k-1*-dimension Euclidean space, where each vertex represents one of the categories. Discrimination is phrased as a regression problem involving $\epsilon-$insensitive residuals and a L2 quadratic ("ridge") penalty on the coefficients of the linear predictors. The objective function can by minimized by a primal Majorization-Minimization (MM) algorithm that

1. relies on quadratic majorization and iteratively re-weighted least squares,
2. is simpler to program than algorithms that pass to the dual of the original optimization problem, and
3. can be accelerated by step doubling.

Comparisons on real and simulated data suggest that the MM algorithm for VDA is competitive in statistical accuracy and computational speed with the best currently available algorithms for discriminant analysis, such as linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), *k*-nearest neighbor, one-vs-rest binary support vector machines, multicategory support vector machines, classification and regression tree (CART), and random forest prediction.

Value

feature	Feature matrix `x` with an intercept vector added as the first column. All entries in the first column should equal 1.
stand.feature	The feature matrix where the all columns are standardized, with the exception of the intercept column which is left unstandardized.
class	Class vector `y`. All elements should be integers between 1 and `classes`.
cases	Number of cases, *n*.
classes	Number of classes, *k*.
features	Number of feautres, *p*.
lambda	Tuning constant `lambda` that was used during analysis.
predicted	Vector of predicted category values based on VDA.
coefficient	The estimated coefficient matrix where the columns represent the coefficients for each predictor variable corresponding to `k-1` outcome categories. The coefficient matrix is used for classifying new cases.
training_error_rate	The percentage of instances in the training set where the predicted outcome category is not equal to the case's true category.
call	The matched call
attr(,"class")	The function results in an object of class "vda.r"

Author(s)

Edward Grant, Xia Li, Kenneth Lange, Tong Tong Wu

Maintainer: Edward Grant <edward.m.grant@gmail.com>

References

Lange, K. and Wu, T.T. (2008) An MM Algorithm for Multicategory Vertex Discriminant Analysis. Journal of Computational and Graphical Statistics, Volume 17, No 3, 527-544.

See Also

For determining the optimal values for lambda, refer to cv.vda.r.

For high-dimensional setting and conduct variable selection, please refer to vda.le.

Examples

```
# load zoo data
# column 1 is name, columns 2:17 are features, column 18 is class
data(zoo)

#matrix containing all predictor vectors
x <- zoo[,2:17]

#outcome class vector
y <- zoo[,18]

#run VDA
out <- vda.r(x, y)

#Predict five cases based on VDA
fivecases <- matrix(0,5,16)
fivecases[1,] <- c(1,0,0,1,0,0,0,1,1,1,0,0,4,0,1,0)
fivecases[2,] <- c(1,0,0,1,0,0,1,1,1,1,0,0,4,1,0,1)
fivecases[3,] <- c(0,1,1,0,1,0,0,0,1,1,0,0,2,1,1,0)
fivecases[4,] <- c(0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0)
fivecases[5,] <- c(0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0)
predict(out, fivecases)
```

---

zoo *zoo data*

Description

The zoo data is a set from the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/). This dataset contains 16 attributes, and 7 animal classes. The first column gives as descriptve name for each case. The next 16 columns each correspond to one feature. The last column is the classification information. The classification is a breakdown of which animals are in which of the 7 types.

Usage

```
data(zoo)
```

Format

[,1]	name: unique for each case
[,2]	hair: Boolean
[,3]	feathers: Boolean
[,4]	eggs: Boolean
[,5]	milk: Boolean
[,6]	airborne: Boolean
[,7]	aquatic: Boolean
[,8]	predator: Boolean
[,9]	toothed: Boolean
[,10]	backbone: Boolean
[,11]	breathes: Boolean
[,12]	venomous: Boolean
[,13]	fins: Boolean
[,14]	legs. Set of values: [0,2,4,5,6,8]
[,15]	tail: Boolean
[,16]	domestic: Boolean
[,17]	catsize: Boolean
[,18]	Class labels, integer values in range [1,7].

Details

There are 7 classes all together:

1. aardvark, antelope, bear, boar, buffalo, calf, cavy, cheetah, deer, dolphin, elephant, fruitbat, giraffe, girl, goat, gorilla, hamster, hare, leopard, lion, lynx, mink, mole, mongoose, opossum, oryx, platypus, polecat, pony, porpoise, puma, pussycat, raccoon, reindeer, seal, sealion, squirrel, vampire, vole, wallaby,wolf
2. chicken, crow, dove, duck, flamingo, gull, hawk, kiwi, lark, ostrich, parakeet, penguin, pheasant, rhea, skimmer, skua, sparrow, swan, vulture, wren
3. pitviper, seasnake, slowworm, tortoise, tuatara
4. bass, carp, catfish, chub, dogfish, haddock, herring, pike, piranha, seahorse, sole, stingray, tuna
5. frog1, frog2, newt, toad
6. flea, gnat, honeybee, housefly, ladybird, moth, termite, wasp
7. clam, crab, crayfish, lobster, octopus, scorpion, seawasp, slug, starfish, worm

Source

Forsyth, R. (1990). UCI Machine Learning Repository - Zoo Data Set [http://archive.ics.uci.edu/ml/datasets/zoo]. Irvine, CA: University of California, School of Information and Computer Science.

Index