Package 'mda'

July 2, 2014

Version 0.4-4

Date 2013-08-16

Author S original by Trevor Hastie & Robert Tibshirani. Original R port by Friedrich Leisch, Kurt Hornik and Brian D. Ripley.

Maintainer Trevor Hastie <hastie@stanford.edu>

Description Mixture and flexible discriminant analysis, multivariate adaptive regression splines (MARS), BRUTO, ...

Title Mixture and flexible discriminant analysis

Depends R (>= 1.9.0), stats, class

Suggests earth

License GPL-2

Repository CRAN

Date/Publication 2013-08-17 07:15:11

NeedsCompilation yes

R topics documented:

`bruto` *Fit an Additive Spline Model by Adaptive Backfitting*

Description

Fit an additive spline model by adaptive backfitting.

Usage

```
bruto(x, y, w, wp, dfmax, cost, maxit.select, maxit.backfit,
      thresh = 0.0001, trace.bruto = FALSE, start.linear = TRUE,
      fit.object, ...)
```

Arguments

`x`	a matrix of numeric predictors (does not include the column of 1s).
`y`	a vector or matrix of responses.
`w`	optional observation weight vector.
`wp`	optional weight vector for each column of `y`; the RSS and GCV criteria use a weighted sum of squared residuals.
`dfmax`	a vector of maximum df (degrees of freedom) for each term.
`cost`	cost per degree of freedom; default is 2.
`maxit.select`	maximum number of iterations during the selection stage.
`maxit.backfit`	maximum number of iterations for the final backfit stage (with fixed lambda).
`thresh`	convergence threshold (default is 0.0001); iterations cease when the relative change in GCV is below this threshold.
`trace.bruto`	logical flag. If `TRUE` (default) a progress report is printed during the fitting.
`start.linear`	logical flag. If `TRUE` (default), the model starts with the linear fit.
`fit.object`	This the object returned by `bruto()`; if supplied, the same model is fit to the presumably new `y`.
`...`	further arguments to be passed to or from methods.

Value

A multiresponse additive model fit object of class "bruto" is returned. The model is fit by adaptive backfitting using smoothing splines. If there are np columns in y, then np additive models are fit, but the same amount of smoothing (df) is used for each term. The procedure chooses between df = 0 (term omitted), df = 1 (term linear) or df > 0 (term fitted by smoothing spline). The model selection is based on an approximation to the GCV criterion, which is used at each step of the backfitting procedure. Once the selection process stops, the model is backfit using the chosen amount of smoothing.

A bruto object has the following components of interest:

lambda	a vector of chosen smoothing parameters, one for each column of x.
df	the df chosen for each column of x.
type	a factor with levels "excluded", "linear" or "smooth", indicating the status of each column of x.
gcv.select gcv.backfit df.select	The sequence of gcv values and df selected during the execution of the function.
nit	the number of iterations used.
fitted.values	a matrix of fitted values.
residuals	a matrix of residuals.
call	the call that produced this object.

References

Trevor Hastie and Rob Tibshirani, *Generalized Additive Models*, Chapman and Hall, 1990 (page 262).

Trevor Hastie, Rob Tibshirani and Andreas Buja "Flexible Discriminant Analysis by Optimal Scoring" JASA 1994, 89, 1255-1270.

See Also

predict.bruto

Examples

```
data(trees)
fit1 <- bruto(trees[,-3], trees[3])
fit1$type
fit1$df
## examine the fitted functions
par(mfrow=c(1,2), pty="s")
Xp <- matrix(sapply(trees[1:2], mean), nrow(trees), 2, byrow=TRUE)
for(i in 1:2) {
  xr <- sapply(trees, range)
  Xp1 <- Xp; Xp1[,i] <- seq(xr[1,i], xr[2,i], len=nrow(trees))
  Xf <- predict(fit1, Xp1)
  plot(Xp1[ ,i], Xf, xlab=names(trees)[i], ylab="", type="l")
}
```

coef.fda *Produce coefficients for an fda or mda object*

Description

a method for coef for extracting the canonical coefficients from an fda or mda object

Usage

```
## S3 method for class 'fda'
coef(object, ...)
```

Arguments

object	an fda or mda object.
...	not relevant

Details

See the references for details.

Value

A coefficient matrix

Author(s)

Trevor Hastie and Robert Tibshirani

References

"Flexible Disriminant Analysis by Optimal Scoring" by Hastie, Tibshirani and Buja, 1994, JASA, 1255-1270.

"Penalized Discriminant Analysis" by Hastie, Buja and Tibshirani, 1995, Annals of Statistics, 73-102.

"Elements of Statisical Learning - Data Mining, Inference and Prediction" (2nd edition, Chapter 12) by Hastie, Tibshirani and Friedman, 2009, Springer

See Also

predict.fda, plot.fda, mars, bruto, polyreg, softmax, confusion,

Examples

```
data(iris)
irisfit <- fda(Species ~ ., data = iris)
coef(irisfit)
mfit=mda(Species~.,data=iris,subclass=2)
coef(mfit)
```

`confusion` *Confusion Matrices*

Description

Compute the confusion matrix between two factors, or for an fda or mda object.

Usage

```
## Default S3 method:
confusion(object, true, ...)
## S3 method for class 'fda'
confusion(object, data, ...)
```

Arguments

`object`	the predicted factor, or an fda or mda model object.
`true`	the true factor.
`data`	a data frame (list) containing the test data.
`...`	further arguments to be passed to or from methods.

Details

This is a generic function.

Value

For the default method essentially `table(object, true)`, but with some useful attribute(s).

See Also

`fda`, `predict.fda`

Examples

```
data(iris)
irisfit <- fda(Species ~ ., data = iris)
confusion(predict(irisfit, iris), iris$Species)
##            Setosa Versicolor Virginica
##     Setosa     50          0         0
## Versicolor      0         48         1
##  Virginica      0          2        49
## attr(, "error"):
## [1] 0.02
```

ESL.mixture *Mixture example from "Elements of Statistical Learning"*

Description

A list with training data and other details for the mixture example

Usage

```
data(ESL.mixture)
```

Format

This list contains the following elements:

x a 200x2 matrix of predictors.

y a 200 vector of y values taking values 0 or 1.

xnew a 6831x2 matrix of prediction points, on a 69x99 grid.

prob a vector of 6831 probabilities - the true probabilities of a 1 at each point in `xnew`.

marginal the marginal distribution of the predictors t each point in `xnew`.

px1 grid values for first coordinate in `xnew`.

px2 grid values for second coordinate in `xnew`.

means a 20 x 2 matrix of means used in the generation of these data.

Source

"Elements of Statistical Learning (second edition)", Hastie, T., Tibshirani, R. and Friedman, J. (2009), Springer, New York. http://www-stat.stanford.edu/ElemStatLearn

fda *Flexible Discriminant Analysis*

Description

Flexible discriminant analysis.

Usage

```
fda(formula, data, weights, theta, dimension, eps, method,
    keep.fitted, ...)
```

Arguments

formula	of the form y~x it describes the response and the predictors. The formula can be more complicated, such as y~log(x)+z etc (see formula for more details). The response should be a factor representing the response variable, or any vector that can be coerced to such (such as a logical variable).
data	data frame containing the variables in the formula (optional).
weights	an optional vector of observation weights.
theta	an optional matrix of class scores, typically with less than J-1 columns.
dimension	The dimension of the solution, no greater than J-1, where J is the number classes. Default is J-1.
eps	a threshold for small singular values for excluding discriminant variables; default is .Machine$double.eps.
method	regression method used in optimal scaling. Default is linear regression via the function polyreg, resulting in linear discriminant analysis. Other possibilities are mars and bruto. For Penalized Discriminant analysis gen.ridge is appropriate.
keep.fitted	a logical variable, which determines whether the (sometimes large) component "fitted.values" of the fit component of the returned fda object should be kept. The default is TRUE if n * dimension < 5000.
...	additional arguments to method.

Value

an object of class "fda". Use predict to extract discriminant variables, posterior probabilities or predicted class memberships. Other extractor functions are coef, confusion and plot.

The object has the following components:

percent.explained
: the percent between-group variance explained by each dimension (relative to the total explained.)

values	optimal scaling regression sum-of-squares for each dimension (see reference). The usual discriminant analysis eigenvalues are given by values / (1-values), which are used to define percent.explained.
means	class means in the discriminant space. These are also scaled versions of the final theta's or class scores, and can be used in a subsequent call to fda (this only makes sense if some columns of theta are omitted—see the references).
theta.mod	(internal) a class scoring matrix which allows predict to work properly.
dimension	dimension of discriminant space.
prior	class proportions for the training data.
fit	fit object returned by method.
call	the call that created this object (allowing it to be update-able)
confusion	confusion matrix when classifying the training data.

The method functions are required to take arguments x and y where both can be matrices, and should produce a matrix of fitted.values the same size as y. They can take additional arguments weights and should all have a ... for safety sake. Any arguments to method can be passed on via the ... argument of fda. The default method polyreg has a degree argument which allows polynomial regression of the required total degree. See the documentation for predict.fda for further requirements of method. The package earth is suggested for this package as well; earth is a more detailed implementation of the mars model, and works as a method argument.

Author(s)

Trevor Hastie and Robert Tibshirani

References

"Flexible Disriminant Analysis by Optimal Scoring" by Hastie, Tibshirani and Buja, 1994, JASA, 1255-1270.

"Penalized Discriminant Analysis" by Hastie, Buja and Tibshirani, 1995, Annals of Statistics, 73-102.

"Elements of Statisical Learning - Data Mining, Inference and Prediction" (2nd edition, Chapter 12) by Hastie, Tibshirani and Friedman, 2009, Springer

See Also

predict.fda, plot.fda, mars, bruto, polyreg, softmax, confusion,

Examples

```
data(iris)
irisfit <- fda(Species ~ ., data = iris)
irisfit
## fda(formula = Species ~ ., data = iris)
##
## Dimension: 2
##
## Percent Between-Group Variance Explained:
```

```
##     v1     v2
##  99.12 100.00
##
## Degrees of Freedom (per dimension): 5
##
## Training Misclassification Error: 0.02 ( N = 150 )

confusion(irisfit, iris)
##            Setosa Versicolor Virginica
##     Setosa     50          0         0
## Versicolor      0         48         1
##  Virginica      0          2        49
## attr(, "error"):
## [1] 0.02

plot(irisfit)

coef(irisfit)
##           [,1]        [,2]
## [1,] -2.126479 -6.72910343
## [2,] -0.837798  0.02434685
## [3,] -1.550052  2.18649663
## [4,]  2.223560 -0.94138258
## [5,]  2.838994  2.86801283

marsfit <- fda(Species ~ ., data = iris, method = mars)
marsfit2 <- update(marsfit, degree = 2)
marsfit3 <- update(marsfit, theta = marsfit$means[, 1:2])
## this refits the model, using the fitted means (scaled theta's)
## from marsfit to start the iterations
```

`gen.ridge` *Penalized Regression*

Description

Perform a penalized regression, as used in penalized discriminant analysis.

Usage

```
gen.ridge(x, y, weights, lambda=1, omega, df, ...)
```

Arguments

`x, y, weights`	the x and y matrix and possibly a weight vector.
`lambda`	the shrinkage penalty coefficient.
`omega`	a penalty object; omega is the eigendecomposition of the penalty matrix, and need not have full rank. By default, standard ridge is used.

df	an alternative way to prescribe lambda, using the notion of equivalent degrees of freedom.
...	currently not used.

Value

A generalized ridge regression, where the coefficients are penalized according to omega. See the function definition for further details. No functions are provided for producing one dimensional penalty objects (omega). `laplacian()` creates a two-dimensional penalty object, suitable for (small) images.

See Also

`laplacian`

`glass` *Glass Identification Database*

Description

The `glass` data frame has 214 observations and 10 variables, representing glass fragments.

Usage

```
data(glass)
```

Format

This data frame contains the following columns:

RI refractive index

Na weight percent in corresponding oxide

Mg weight percent in corresponding oxide

Al weight percent in corresponding oxide

Si weight percent in corresponding oxide

K weight percent in corresponding oxide

Ca weight percent in corresponding oxide

Ba weight percent in corresponding oxide

Fe weight percent in corresponding oxide

Type Type of glass:

- **1** building_windows_float_processed,
- **2** building_windows_non_float_processed,
- **3** vehicle_windows_float_processed,
- **4** vehicle_windows_non_float_processed (none in this database),
- **5** containers,
- **6** tableware,
- **7** headlamps

Source

P. M. Murphy and D. W. Aha (1999), UCI Repository of Machine Learning Databases, ftp://ics.uci.edu/pub/machine-learning-databases

---

`laplacian` *create penalty object for two-dimensional smoothing.*

Description

Creates a penalty matrix for use by `gen.ridge` for two-dimensional smoothing.

Usage

```
laplacian(size, compose)
laplacian(size = 16, compose = FALSE)
```

Arguments

`size`	dimension of the image is `size x size`; default is 16.
`compose`	default is `compose=FALSE`, which means the penalty is returned as an eigen-decomposition. If `compose=TRUE`, a penalty matrix is returned.

Details

Formulas are used to construct a laplacian for smoothing a square image.

Value

If `compose=FALSE`, an eigen-decomposition object is returned. The `vectors` component is a `size^2 x size^2` orthogonal matrix, and the `$values` component is a `size^2` vector of non-negative eigen-values. If `compose=TRUE`, these are multiplied together to form a single matrix.

Author(s)

Trevor Hastie <hastie@stanford.edu

References

Here we follow very closely the material on page 635 in JASA 1991 of O'Sullivan's article on discretized Laplacian Smoothing

See Also

`gen.ridge`,`fda`

mars *Multivariate Adaptive Regression Splines*

Description

Multivariate adaptive regression splines.

Usage

```
mars(x, y, w, wp, degree, nk, penalty, thresh, prune, trace.mars,
     forward.step, prevfit, ...)
```

Arguments

x	a matrix containing the independent variables.
y	a vector containing the response variable, or in the case of multiple responses, a matrix whose columns are the response values for each variable.
w	an optional vector of observation weights (currently ignored).
wp	an optional vector of response weights.
degree	an optional integer specifying maximum interaction degree (default is 1).
nk	an optional integer specifying the maximum number of model terms.
penalty	an optional value specifying the cost per degree of freedom charge (default is 2).
thresh	an optional value specifying forward stepwise stopping threshold (default is 0.001).
prune	an optional logical value specifying whether the model should be pruned in a backward stepwise fashion (default is TRUE).
trace.mars	an optional logical value specifying whether info should be printed along the way (default is FALSE).
forward.step	an optional logical value specifying whether forward stepwise process should be carried out (default is TRUE).
prevfit	optional data structure from previous fit. To see the effect of changing the penalty parameter, one can use prevfit with forward.step = FALSE.
...	further arguments to be passed to or from methods.

Value

An object of class "mars", which is a list with the following components:

call	call used to mars.
all.terms	term numbers in full model. 1 is the constant term. Remaining terms are in pairs (2 3, 4 5, and so on). all.terms indicates nonsingular set of terms.
selected.terms	term numbers in selected model.
penalty	the input penalty value.

degree	the input degree value.
thresh	the input threshold value.
gcv	gcv of chosen model.
factor	matrix with ij-th element equal to 1 if term i has a factor of the form $x_j > c$, equal to -1 if term i has a factor of the form $x_j \leq c$, and to 0 if xj is not in term i.
cuts	matrix with ij-th element equal to the cut point c for variable j in term i.
residuals	residuals from fit.
fitted	fitted values from fit.
lenb	length of full model.
coefficients	least squares coefficients for final model.
x	a matrix of basis functions obtained from the input x matrix.

Note

This function was coded from scratch, and did not use any of Friedman's mars code. It gives quite similar results to Friedman's program in our tests, but not exactly the same results. We have not implemented Friedman's anova decomposition nor are categorical predictors handled properly yet. Our version does handle multiple response variables, however.

Author(s)

Trevor Hastie and Robert Tibshirani

References

J. Friedman, "Multivariate Adaptive Regression Splines" (with discussion) (1991). *Annals of Statistics*, **19**/1, 1–141.

See Also

predict.mars, model.matrix.mars.

Package **earth** also provides multivariate adaptive regression spline models based on the Hastie/Tibshirani mars code in package **mda**, adding some extra features. It can be used in the method argument of fda or mda.

Examples

```
data(trees)
fit1 <- mars(trees[,-3], trees[3])
showcuts <- function(obj)
{
  tmp <- obj$cuts[obj$sel, ]
  dimnames(tmp) <- list(NULL, names(trees)[-3])
  tmp
}
showcuts(fit1)
```

```
## examine the fitted functions
par(mfrow=c(1,2), pty="s")
Xp <- matrix(sapply(trees[1:2], mean), nrow(trees), 2, byrow=TRUE)
for(i in 1:2) {
  xr <- sapply(trees, range)
  Xp1 <- Xp; Xp1[,i] <- seq(xr[1,i], xr[2,i], len=nrow(trees))
  Xf <- predict(fit1, Xp1)
  plot(Xp1[ ,i], Xf, xlab=names(trees)[i], ylab="", type="l")
}
```

---

mda *Mixture Discriminant Analysis*

Description

Mixture discriminant analysis.

Usage

```
mda(formula, data, subclasses, sub.df, tot.df, dimension, eps,
    iter, weights, method, keep.fitted, trace, ...)
```

Arguments

formula	of the form y~x it describes the response and the predictors. The formula can be more complicated, such as y~log(x)+z etc (see formula for more details). The response should be a factor representing the response variable, or any vector that can be coerced to such (such as a logical variable).
data	data frame containing the variables in the formula (optional).
subclasses	Number of subclasses per class, default is 3. Can be a vector with a number for each class.
sub.df	If subclass centroid shrinking is performed, what is the effective degrees of freedom of the centroids per class. Can be a scalar, in which case the same number is used for each class, else a vector.
tot.df	The total df for all the centroids can be specified rather than separately per class.
dimension	The dimension of the reduced model. If we know our final model will be confined to a discriminant subspace (of the subclass centroids), we can specify this in advance and have the EM algorithm operate in this subspace.
eps	A numerical threshold for automatically truncating the dimension.
iter	A limit on the total number of iterations, default is 5.
weights	*NOT* observation weights! This is a special weight structure, which for each class assigns a weight (prior probability) to each of the observations in that class of belonging to one of the subclasses. The default is provided by a call to mda.start(x, g, subclasses, trace, ...) (by this time x and g are

known). See the help for mda.start. Arguments for mda.start can be provided via the ... argument to mda, and the weights argument need never be accessed. A previously fit mda object can be supplied, in which case the final subclass responsibility weights are used for weights. This allows the iterations from a previous fit to be continued.

method — regression method used in optimal scaling. Default is linear regression via the function polyreg, resulting in the usual mixture model. Other possibilities are mars and bruto. For penalized mixture discriminant models gen.ridge is appropriate.

keep.fitted — a logical variable, which determines whether the (sometimes large) component "fitted.values" of the fit component of the returned mda object should be kept. The default is TRUE if n * dimension < 5000.

trace — if TRUE, iteration information is printed. Note that the deviance reported is for the posterior class likelihood, and not the full likelihood, which is used to drive the EM algorithm under mda. In general the latter is not available.

... — additional arguments to mda.start and to method.

Value

An object of class c("mda", "fda"). The most useful extractor is predict, which can make many types of predictions from this object. It can also be plotted, and any functions useful for fda objects will work here too, such as confusion and coef.

The object has the following components:

percent.explained — the percent between-group variance explained by each dimension (relative to the total explained.)

values — optimal scaling regression sum-of-squares for each dimension (see reference).

means — subclass means in the discriminant space. These are also scaled versions of the final theta's or class scores, and can be used in a subsequent call to mda (this only makes sense if some columns of theta are omitted—see the references)

theta.mod — (internal) a class scoring matrix which allows predict to work properly.

dimension — dimension of discriminant space.

sub.prior — subclass membership priors, computed in the fit. No effort is currently spent in trying to keep these above a threshold.

prior — class proportions for the training data.

fit — fit object returned by method.

call — the call that created this object (allowing it to be update-able).

confusion — confusion matrix when classifying the training data.

weights — These are the subclass membership probabilities for each member of the training set; see the weights argument.

assign.theta — a pointer list which identifies which elements of certain lists belong to individual classes.

`deviance` The multinomial log-likelihood of the fit. Even though the full log-likelihood drives the iterations, we cannot in general compute it because of the flexibility of the `method` used. The deviance can increase with the iterations, but generally does not.

The `method` functions are required to take arguments `x` and `y` where both can be matrices, and should produce a matrix of `fitted.values` the same size as `y`. They can take additional arguments `weights` and should all have a `...` for safety sake. Any arguments to method() can be passed on via the `...` argument of `mda`. The default method `polyreg` has a `degree` argument which allows polynomial regression of the required total degree. See the documentation for `predict.fda` for further requirements of `method`. The package `earth` is suggested for this package as well; `earth` is a more detailed implementation of the mars model, and works as a `method` argument.

The function `mda.start` creates the starting weights; it takes additional arguments which can be passed in via the `...` argument to `mda`. See the documentation for `mda.start`.

Author(s)

Trevor Hastie and Robert Tibshirani

References

"Flexible Disriminant Analysis by Optimal Scoring" by Hastie, Tibshirani and Buja, 1994, JASA, 1255-1270.

"Penalized Discriminant Analysis" by Hastie, Buja and Tibshirani, 1995, Annals of Statistics, 73-102

"Discriminant Analysis by Gaussian Mixtures" by Hastie and Tibshirani, 1996, JRSS-B, 155-176.

"Elements of Statisical Learning - Data Mining, Inference and Prediction" (2nd edition, Chapter 12) by Hastie, Tibshirani and Friedman, 2009, Springer

See Also

`predict.mda`, `mars`, `bruto`, `polyreg`, `gen.ridge`, `softmax`, `confusion`

Examples

```
data(iris)
irisfit <- mda(Species ~ ., data = iris)
irisfit
## Call:
## mda(formula = Species ~ ., data = iris)
##
## Dimension: 4
##
## Percent Between-Group Variance Explained:
##     v1     v2     v3     v4
##  96.02  98.55  99.90 100.00
##
## Degrees of Freedom (per dimension): 5
##
## Training Misclassification Error: 0.02 ( N = 150 )
```

```
##
## Deviance: 15.102

data(glass)
# random sample of size 100
samp <- c(1, 3, 4, 11, 12, 13, 14, 16, 17, 18, 19, 20, 27, 28, 31,
          38, 42, 46, 47, 48, 49, 52, 53, 54, 55, 57, 62, 63, 64, 65,
          67, 68, 69, 70, 72, 73, 78, 79, 83, 84, 85, 87, 91, 92, 94,
          99, 100, 106, 107, 108, 111, 112, 113, 115, 118, 121, 123,
          124, 125, 126, 129, 131, 133, 136, 139, 142, 143, 145, 147,
          152, 153, 156, 159, 160, 161, 164, 165, 166, 168, 169, 171,
          172, 173, 174, 175, 177, 178, 181, 182, 185, 188, 189, 192,
          195, 197, 203, 205, 211, 212, 214)
glass.train <- glass[samp,]
glass.test <- glass[-samp,]
glass.mda <- mda(Type ~ ., data = glass.train)
predict(glass.mda, glass.test, type="post") # abbreviations are allowed
confusion(glass.mda,glass.test)
```

`mda.start` *Initialization for Mixture Discriminant Analysis*

Description

Provide starting weights for the `mda` function which performs discriminant analysis by gaussian mixtures.

Usage

```
mda.start(x, g, subclasses = 3, trace.mda.start = FALSE,
          start.method = c("kmeans", "lvq"), tries = 5,
          criterion = c("misclassification", "deviance"), ...)
```

Arguments

`x`	The x data, or an mda object.
`g`	The response vector g.
`subclasses`	number of subclasses per class, as in `mda`.
`trace.mda.start`	Show results of each iteration.
`start.method`	Either `"kmeans"` or `"lvq"`. The latter requires package **class** (from the **VR** package bundle.
`tries`	Number of random starts.
`criterion`	By default, classification errors on the training data. Posterior deviance is also an option.
`...`	arguments to be passed to the mda fitter when using posterior deviance.

Value

A list of weight matrices, one for each class.

`model.matrix.mars` *Produce a Design Matrix from a 'mars' Object*

Description

Produce a design matrix from a 'mars' object.

Usage

```
## S3 method for class 'mars'
model.matrix(object, x, which, full = FALSE, ...)
```

Arguments

`object`	a mars object.
`x`	optional argument; if supplied, the mars basis functions are evaluated at these new observations.
`which`	which columns should be used. The default is to use the columns described by the component `selected.terms` on `object`.
`full`	if `TRUE` the entire set of columns are selected, even redundant ones. This is used for updating a mars fit.
`...`	further arguments to be passed from or to methods.

Value

A model matrix corresponding to the selected columns.

See Also

`mars`, `predict.mars`

plot.fda *Plot for Flexible Discriminant Analysis*

Description

Plot in discriminant (canonical) coordinates a fda or (by inheritance) a mda object.

Usage

```
## S3 method for class 'fda'
plot(x, data, coords, group, colors, pch, mcolors, mpch, pcex, mcex, ...)
```

Arguments

x	an object of class "fda".
data	the data to plot in the discriminant coordinates. If group="true", then data should be a data frame with the same variables that were used in the fit. If group="predicted", data need not contain the response variable, and can in fact be the correctly-sized "x" matrix.
coords	vector of coordinates to plot, with default coords="c(1,2)". All pairs of plots are produced.
group	if group="true" (the default), each point is color and symbol coded according to the response in data. If group="predicted", the class of each point is predicted from the model, and used instead.
colors	a vector of colors to be used in the plotting.
pch	a vector of plotting characters.
mcolors	a vector of colors for the class centroids; default is colors.
mpch	a vector of plotting characters for the centroids.
pcex	character expansion factor for the points; defualt is pcex="0.5".
mcex	character expansion factor for the centroids; defualt is pcex="2.5".
...	further arguments to be passed to or from methods.

See Also

fda, mda, predict.fda

Examples

```
data(iris)
irisfit <- fda(Species ~ ., data = iris)
plot(irisfit)
data(ESL.mixture)
## Not a data frame
mixture.train=ESL.mixture[c("x","y")]
mixfit=mda(y~x, data=mixture.train)
plot(mixfit, mixture.train)
plot(mixfit, data=ESL.mixture$xnew, group="pred")
```

polyreg *Polynomial Regression*

Description

Simple minded polynomial regression.

Usage

```
polyreg(x, y, w, degree = 1, monomial = FALSE, ...)
```

Arguments

x	predictor matrix.
y	response matrix.
w	optional (positive) weights.
degree	total degree of polynomial basis (default is 1).
monomial	If TRUE a monomial basis is used (no cross terms). Default is FALSE.
...	currently not used.

Value

A polynomial regression fit, containing the essential ingredients for its predict method.

predict.bruto *Predict method for BRUTO Objects*

Description

Predicted values based on 'bruto' additive spline models which are fit by adaptive backfitting.

Usage

```
## S3 method for class 'bruto'
predict(object, newdata, type=c("fitted", "terms"), ...)
```

Arguments

object	a fitted bruto object
newdata	values at which predictions are to be made.
type	if type is "fitted", the fitted values are returned. If type is "terms", a list of fitted terms is returned, each with an x and y component. These can be used to show the fitted functions.
...	further arguments to be passed to or from methods.

Value

Either a fit matrix or a list of fitted terms.

See Also

bruto, predict

Examples

```
data(trees)
fit1 <- bruto(trees[,-3], trees[3])
fitted.terms <- predict(fit1, as.matrix(trees[,-3]), type = "terms")
par(mfrow=c(1,2), pty="s")
for(tt in fitted.terms) plot(tt, type="l")
```

---

predict.fda *Classify by Flexible Discriminant Analysis*

Description

Classify observations in conjunction with fda.

Usage

```
## S3 method for class 'fda'
predict(object, newdata, type, prior, dimension, ...)
```

Arguments

object	an object of class "fda".
newdata	new data at which to make predictions. If missing, the training data is used.
type	kind of predictions: type = "class" (default) produces a fitted factor, type = "variates" produces a matrix of discriminant (canonical) variables, type = "posterior" produces a matrix of posterior probabilities (based on a gaussian assumption), and type = "hierarchical" produces the predicted class in sequence for models of all dimensions.
prior	the prior probability vector for each class; the default is the training sample proportions.
dimension	the dimension of the space to be used, no larger than the dimension component of object.
...	further arguments to be passed to or from methods.

Value

An appropriate object depending on type. object has a component fit which is regression fit produced by the method argument to fda. There should be a predict method for this object which is invoked. This method should itself take as input object and optionally newdata.

See Also

fda, mars, bruto, polyreg, softmax, confusion

Examples

```
data(iris)
irisfit <- fda(Species ~ ., data = iris)
irisfit
## Call:
## fda(x = iris$x, g = iris$g)
##
## Dimension: 2
##
## Percent Between-Group Variance Explained:
##     v1  v2
##  99.12 100
confusion(predict(irisfit, iris), iris$Species)
##            Setosa Versicolor Virginica
##     Setosa     50          0         0
## Versicolor      0         48         1
##  Virginica      0          2        49
## attr(, "error"):
## [1] 0.02
```

---

predict.mars *Predict method for MARS Objects*

Description

Predicted values based on 'mars' multivariate adaptive regression spline models.

Usage

```
## S3 method for class 'mars'
predict(object, newdata, ...)
```

Arguments

object	an object of class "mars".
newdata	values at which predictions are to be made.
...	further arguments to be passed to or from methods.

Value

the fitted values.

See Also

mars, predict, model.matrix.mars

predict.mda *Classify by Mixture Discriminant Analysis*

Description

Classify observations in conjunction with mda.

Usage

```
## S3 method for class 'mda'
predict(object, newdata, type, prior, dimension, g, ...)
```

Arguments

object	a fitted mda object.
newdata	new data at which to make predictions. If missing, the training data is used.
type	kind of predictions: type = "class" (default) produces a fitted factor, type = "variates" produces a matrix of discriminant variables (note that the maximal dimension is determined by the number of subclasses), type = "posterior" produces a matrix of posterior probabilities (based on a gaussian assumption), type = "hierarchical" produces the predicted class in sequence for models of dimensions specified by dimension argument.
prior	the prior probability vector for each class; the default is the training sample proportions.
dimension	the dimension of the space to be used, no larger than the dimension component of object, and in general less than the number of subclasses. dimension can be a vector for use with type = "hierarchical".
g	???
...	further arguments to be passed to or from methods.

Value

An appropriate object depending on type. object has a component fit which is regression fit produced by the method argument to mda. There should be a predict method for this object which is invoked. This method should itself take as input object and optionally newdata.

See Also

mda, fda, mars, bruto, polyreg, softmax, confusion

Examples

```
data(glass)
samp <- sample(1:nrow(glass), 100)
glass.train <- glass[samp,]
glass.test <- glass[-samp,]
glass.mda <- mda(Type ~ ., data = glass.train)
predict(glass.mda, glass.test, type = "post") # abbreviations are allowed
confusion(glass.mda, glass.test)
```

---

softmax *Find the Maximum in Each Row of a Matrix*

Description

Find the maximum in each row of a matrix.

Usage

```
softmax(x, gap = FALSE)
```

Arguments

x	a numeric matrix.
gap	if TRUE, the difference between the largest and next largest column is returned.

Value

A factor with levels the column labels of x and values the columns corresponding to the maximum column. If gap = TRUE a list is returned, the second component of which is the difference between the largest and next largest column of x.

See Also

predict.fda, confusion, fda mda

Examples

```
data(iris)
irisfit <- fda(Species ~ ., data = iris)
posteriors <- predict(irisfit, type = "post")
confusion(softmax(posteriors), iris[, "Species"])
```

Index