Fitting Multivariate Adaptive Regression Splines

Description

`mars` is used to fit multivariate adaptive regression splines.

Usage

```
mars(formula, data, control = NULL)
```

Arguments

`formula`	an object of class "formula", specifying the response and explanatory variables.
`data`	a data frame that containing that variables in the model.
`control`	an optional argument of class "control" that contains model specification parameters `Mmax` (maximum number of splines), `d` (smoothing parameter for `lof` function), and `trace` (controls printing of fitting process details).

Details

Models for `mars` are specified with the formula object. A model should have the form `response ~ terms` where `response` is the (numeric) response vector and `terms` is the series of terms which will be used to model the response. The formation of the formula object follows standard formula rules. A terms specification of the form `first + second` indicates all the terms in `first` together with all the terms in `second`. The form `first:second` indicates the interactions of all terms in `first` with all terms in `second`. The specification `first*second` indicates the *cross* of `first` and `second`, which is `first + second + first:second`.

If the user wishes to specify their own control, it should be specified with the `mars.control(Mmax = 2, d = 3, trace = FALSE)` helper function. The resulting output can then be passed to `mars` as `control`. The function `mars.control` calls `validate_mars.control` and `new_mars.control` to validate and construct the control object. `Mmax` specifies the maximum number of splines the user wishes to be constructed, with the minimum permitted being 2. `d` is a smoothing parameter used in the `lof` function, which uses generalized cross-validation to assess lack of fit. `trace = TRUE` will print details of the fitting process.

After creating the model matrix, `mars` calls `fwd_stepwise`, `bwd_stepwise`, and `lm` to produce the details of the model and the mars object. `fwd_stepwise` adds basis functions based on greatest reduction of `lof` to produce the knots of the model. Basis functions are the product of a series of `hinge` functions. To avoid overfitting, `bwd_stepwise` trims the model by deleting terms that have insignificant effect on the model. These procedures mimic those described by Friedman (1991) in "Multivariate Adaptive Regression Splines". The output is then passed to `lm` to obtain the coefficients of the basis functions. These product basis functions and coefficients form the splines of the mars model.

The `mars` class inherits from the `lm` class.

Value

`mars` returns an object of `"mars"`.

The function `summary` is used to obtain and print a summary of the results. Specifically, details of the coefficients, basis functions, hinge functions, splits, and generalized cross-validation information from `lof` can be found. The function `anova.mars` produces a list of the univariate ANOVA contributions and another of the bivariate ANOVA contributions. The generic accessor functions `coefficients`, `effects`, `fitted.values` and `residuals` can also be used to extract information about the value returned by `mars`.

An object of class `"mars"` is a list containing at least the following components:

`call`	the form of the original call of the mars function as a language object.
`formula`	the formula that was used as the argument in the mars call.
`y`	the response of the model, that was specified in the formula.
`B`	a matrix of the basis functions, evaluated at the inputs.
`splits`	a list of all basis functions, summarized as the inputs of the hinge functions that form them.
`data`	the data that was originally provided to the mars function.
`coefficients`	a named vector of coefficients for the basis functions of the model.
`residuals`	the residuals, which is the response minus fitted values.
`effects`	the effects of the model, coming from `lm`.
`rank`	the rank of the model, coming from `lm`.
`fitted.values`	the fitted values resulting from the inputted terms.
`assign`	the assign attribute of the model, coming from `lm`.

`qr`	logicals, coming from `lm`.
`df.residual`	the residual degrees of freedom, coming from `lm`.
`call`	the matched call, coming from `lm`.
`terms`	the `terms` object used, coming from `lm`.
`model`	if requested (the default), the model frame used, coming from `lm`.

Author(s)

The implementation of model was done by Kyle Deng, Andy Liang, and Rachel Loo with aid from Professor Brad McNeney and Pulindu Ratnasekera.

References

Friedman, J. (1991). Multivariate Adaptive Regression Splines. The Annals of Statistics, 19(1), 1-67. Retrieved April 20, 2021, from http://www.jstor.org/stable/2241837

See Also

`summary.mars` for summaries of mars objects (basis functions, hinge information, coefficients and generalized cross-validation information).

`anova.mars` for lists of the univariate and bivariate ANOVA decompositions.

The generic functions `coef`, `effects`, `residuals`, `fitted`, `vcov` are inherited from `lm`.

`predict.mars` for prediction from the model of given data.

`plot.mars` for plots of the univariate ANOVA contribution of a given variable (for numeric input) with red vertical lines marking knots or boxplots by category of residuals for a given variable (for categorical input).

`print.mars` for printing mars objects (function call and coefficients).

Examples

```
require(ISLR)

data(Wage)
# First Test
mc <- mars.control(Mmax=10)
mout <- mars(wage ~ age + education, data=Wage, control=mc)
```

```
ff <- fitted(mout)
p1 <- predict(mout)
p2 <- predict(mout,newdata=data.frame(age=Wage$age,education=Wage$education))
head(cbind(ff,p1,p2)) # columns should be identical
mout # tests print method
summary(mout) #test summary method
anova(mout) # test anova method
plot(mout, "education") # test plot method
print(mout)

# Second test
require(mlbench)
data(BostonHousing)
mc1 <- mars.control(Mmax=8)
mout1 <- mars(medv ~ dis + crim, data = BostonHousing, control = mc1)
ff1 <- fitted(mout1)
p11 <- predict(mout1)
p21 <- predict(mout1,newdata=data.frame(dis = BostonHousing$dis, crim =
BostonHousing$crim))
head(cbind(ff1,p11,p21))
mout1
summary(mout1)
anova(mout1)
plot(mout1)
print(mout1)

# Third Test
data(cars)
mcc <- mars.control(Mmax=3)
mcars <- mars(speed ~ dist, data = cars, control = mcc)
ff2 <- fitted(mcars)
p12 <- predict(mcars)
p22 <- predict(mcars, newdata=data.frame(dist = cars$dist))
print(mcars)
summary(mcars)
anova(mcars)
plot(mcars, "dist", color = "blue")
```