Package: nestedLogit (via r-universe)

September 3, 2024

Title Nested Dichotomy Logistic Regression Models

Version 0.3.3

Date 2024-06-03

Description Provides functions for specifying and fitting nested dichotomy logistic regression models for a multi-category response and methods for summarising and plotting those models. Nested dichotomies are statistically independent, and hence provide an additive decomposition of tests for the overall 'polytomous' response. When the dichotomies make sense substantively, this method can be a simpler alternative to the standard 'multinomial' logistic model which compares response categories to a reference level. See: J. Fox (2016), ``Applied Regression Analysis and Generalized Linear Models'', 3rd Ed., ISBN 1452205663.

License GPL (>=2)

URL https://github.com/friendly/nestedLogit

BugReports https://github.com/friendly/nestedLogit/issues

Depends R (>= 3.5.0)

Imports broom, car, dplyr, effects, graphics, grDevices, stats, stringr, tibble

Suggests carData, geomtextpath, ggplot2, ggeffects, here, lobstr, knitr, nnet, rmarkdown, scales, spelling, testthat, tidyr

VignetteBuilder knitr, rmarkdown

Encoding UTF-8

Language en-US

LazyData TRUE

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Repository https://friendly.r-universe.dev

RemoteUrl https://github.com/friendly/nestedLogit

RemoteRef HEAD

RemoteSha 261d84632db9977e5d995a168f6576e20ddd8b0c

Contents

`as.data.frame.predictNestedLogit`

Convert a Predicted Objects to a data.frame

Description

These functions provide simple ways to convert the results of `predict.nestedLogit` to a data frame in a consistent format for plotting and other actions.

Usage

```
## S3 method for class 'predictNestedLogit'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	a `"predictNestedLogit"` object
`row.names`	row.names for result (for conformity with generic; not currently used)
`optional`	logical. If TRUE, setting row names and converting column names (to syntactic names: see `make.names` is optional
`...`	other arguments (unused)

Value

- For `predict(..., model="nested")` (the default), returns a data frame containing the values of predictors along with the columns `response`, `p`, `se.p`, `logit`, `se.logit`.
- For `predict(..., model="dichotomies")`, returns a data frame containing the values of predictors along with the columns `response`, `logit`, and `se.logit`.

Examples

```
data("Womenlf", package = "carData")
comparisons <- logits(work=dichotomy("not.work", c("parttime", "fulltime")),
                      full=dichotomy("parttime", "fulltime"))

wlf.nested <- nestedLogit(partic ~ hincome + children,
                          dichotomies = comparisons,
                          data=Womenlf)
# get predicted values for a grid of `hincome` and `children`
new <- expand.grid(hincome=seq(0, 45, length=10),
                   children=c("absent", "present"))

pred.nested <- predict(wlf.nested, new)
plotdata <- as.data.frame(pred.nested)
str(plotdata)

# Predicted logit values for the dichotomies
pred.dichot <- predict(wlf.nested, new, model = "dichotomies")
plotlogit <- as.data.frame(pred.dichot)
str(plotlogit)
```

`broomMethods` *Broom Related Methods*

Description

These functions give compact summaries of a `"nestedLogit"` object

`glance` Construct a single row summaries for the dichotomies `"nestedLogit"` model.

`tidy` Summarizes the terms in `"nestedLogit"` model.

Usage

```
## S3 method for class 'nestedLogit'
glance(x, ...)

## S3 method for class 'nestedLogit'
tidy(x, ...)
```

Arguments

`x`	an object of class `"nestedLogit"`.
`...`	arguments to be passed down.

Value

- `glance` returns a `tibble` containing one row of fit statistics for each dichotomy, labeled `response`. See `glance` for details.
- `tidy` returns a `tibble` containing coefficient estimates and test statistics for the combinations of `response` and `term`. See `tidy` for details.

See Also

nestedMethods, glance, tidy

Examples

```
data("Womenlf", package = "carData")
m <-  nestedLogit(partic ~ hincome + children,
                  dichotomies = logits(work=dichotomy("not.work",
                                                      working=c("parttime", "fulltime")),
                                       full=dichotomy("parttime", "fulltime")),
                  data=Womenlf)

# one-line summaries
broom::glance(m)
# coefficients and tests
broom::tidy(m)
```

Effect.nestedLogit *Effect Displays for Nested Logit Models*

Description

Computes effects (in the sense of the **effects** package—see, in particular, Effect)—for "nestedLogit" models, which then can be used with other functions in the **effects** package, for example, predictorEffects and to produce effect plots.

Usage

```
## S3 method for class 'nestedLogit'
Effect(
  focal.predictors,
  mod,
  confidence.level = 0.95,
  fixed.predictors = NULL,
  ...
)
```

Arguments

focal.predictors
: a character vector of the names of one or more of the predictors in the model, for which the effect display should be computed.

mod
: a "nestedLogit" model object.

confidence.level
: for point-wise confidence bands around the effects (the default is 0.95).

`fixed.predictors`
: controls the values at which other predictors are fixed; see `Effect` for details; if `NULL` (the default), numeric predictors are set to their means, factors to their distribution in the data.

`...`
: optional arguments to be passed to the `Effect` method for binary logit models (fit by the `glm` function).

Value

an object of class `"effpoly"` (see `Effect`).

Author(s)

John Fox

References

John Fox and Sanford Weisberg (2019). *An R Companion to Applied Regression*, 3rd Edition. Sage, Thousand Oaks, CA.

John Fox, Sanford Weisberg (2018). Visualizing Fit and Lack of Fit in Complex Regression Models with Predictor Effect Plots and Partial Residuals. *Journal of Statistical Software*, 87(9), 1-27.

See Also

`Effect`, `plot.effpoly`, `predictorEffects`

Examples

```
data("Womenlf", package = "carData")
comparisons <- logits(work=dichotomy("not.work",
                                     working=c("parttime", "fulltime")),
                      full=dichotomy("parttime", "fulltime"))
m <- nestedLogit(partic ~ hincome + children,
                 dichotomies = comparisons,
                 data=Womenlf)
peff.women <- effects::predictorEffects(m)
plot(peff.women)
plot(peff.women, axes=list(y=list(style="stacked")))
summary(peff.women)

dichots <- logits(AB_CD = dichotomy(c("A", "B"), c("C", "D")),
                  A_B   = dichotomy("A", "B"),
                  C_D   = dichotomy("C", "D"))
m.health <- nestedLogit(product4 ~ age + gender*household + position_level,
                        dichotomies = dichots, data = HealthInsurance)
eff.gen.hh <- effects::Effect(c("gender", "household"), m.health,
                              xlevels=list(household=0:7))
eff.gen.hh
plot(eff.gen.hh, axes=list(x=list(rug=FALSE)))
plot(eff.gen.hh, axes=list(x=list(rug=FALSE),
                           y=list(style="stacked")))
```

GSS *Data From the U.S. General Social Survey 1972-2016*

Description

This data set is drawn from the U.S. General Social Survey (GSS) for years between 1972 and 2016.

Usage

```
data("GSS", package = "nestedLogit")
```

Format

A data frame with 44091 rows and 3 columns.

parentdeg A factor representing parents' attained level of education (highest "degree" obtained), recording the higher of mother's and father's education, with levels `"l.t.highschool"`, `"highschool"`, `"college"`, and `"graduate"`.

degree The respondent's level of education, a factor with the same levels as `parentdeg`.

year The year of the survey, between `1972` and `2016`.

Source

General Social Survey, NORC, The University of Chicago https://www.norc.org/Research/Projects/Pages/general-social-survey.aspx.

See Also

`nestedLogit`.

Examples

```
round(100*with(GSS, prop.table(table(degree, parentdeg), 2)))
m.GSS <- nestedLogit(degree ~ parentdeg*year,
                     continuationLogits(c("l.t.highschool",  "highschool",
                                          "college", "graduate")),
                     data=GSS)
car::Anova(m.GSS)
summary(m.GSS)
```

`HealthInsurance` *Choice of Health Insurance Product*

Description

A company recently introduced a new health insurance provider for its employees. At the beginning of the year the employees had to choose one of three (or four) different health plan products from this provider to best suit their needs.

This dataset was modified from its original source (McNulty, 2022) for the present purposes by adding a fourth choice, sampled randomly from the original three.

Usage

```
data("HealthInsurance", package = "nestedLogit")
```

Format

A data frame with 1448 rows and 7 columns.

product Choice among three products, a factor with levels `"A"`, `"B"`, and `"C"`.

product4 Choice among four products, a factor with levels `"A"`, `"B"`, `"C"`, and `"D"`.

age The age of the individual, in years.

household The number of people living with the individual in the same household.

position_level Position level in the company at the time the choice was made, where 1 is is the lowest level and 5 is the highest, a numeric vector.

gender The gender of the individual, a factor with levels `"Female"` and `"Male"`.

absent The number of days the individual was absent from work in the year prior to the choice,

Source

Originally taken from McNulty, K. (2022). *Handbook of Regression Modeling in People Analytics*, https://peopleanalytics-regression-book.org/data/health_insurance.csv.

See Also

`nestedLogit`.

Examples

```
lbinary <- logits(AB_CD = dichotomy(c("A", "B"), c("C", "D")),
                  A_B   = dichotomy("A", "B"),
                  C_D   = dichotomy("C", "D"))
as.matrix(lbinary)
health.nested <- nestedLogit(product4 ~ age  + gender * household + position_level,
                             dichotomies = lbinary, data = HealthInsurance)
                             car::Anova(health.nested)
coef(health.nested)
```

models *Extract Binary Logit Models from a* nestedLogit *Object*

Description

models is used to extract "glm" objects representing binary logit models from a "nestedLogit" object.

Usage

```
models(model, select, as.list = FALSE)

## S3 method for class 'nestedLogit'
models(model, select, as.list = FALSE)
```

Arguments

model	a "nestedLogit" model.
select	a numeric or character vector giving the number(s) or names(s) of one or more binary logit models to be extracted from model; if absent, a list of all of the binary logits models in model is returned.
as.list	if TRUE (the default is FALSE) and one binary logit model is selected, return the "glm" object in a one-element named list; otherwise a single model is returned directly as a "glm" object; when more than one binary logit model is selected, the corresponding "glm" objects are *always* returned as a named list.

Value

model returns either a single "glm" object (see glm) or a list of "glm" objects, each representing a binary logit model.

Examples

```
data("Womenlf", package = "carData")
comparisons <- logits(work=dichotomy("not.work",
                                     working=c("parttime", "fulltime")),
                      full=dichotomy("parttime", "fulltime"))
m <- nestedLogit(partic ~ hincome + children,
                 dichotomies = comparisons,
                 data=Womenlf)

# extract a binomial logit model
models(m, "work")
# use that to plot residuals
plot(density(residuals(models(m, "work"))))
```

nestedHypotheses *Hypothesis-Testing and Related Methods for* "nestedLogit" *Objects*

Description

Various methods for testing hypotheses about nested logit models.

`Anova` Calculates type-II or type-III analysis-of-variance tables for `"nestedLogit"` objects; see `Anova` in the **car** package.

`anova` Computes sequential analysis of variance (or deviance) tables for one or more fitted `"nestedLogit"` objects; see `anova`.

`linearHypothesis` Computes Wald tests for linear hypotheses; see `linearHypothesis` in the **car** package.

`logLik` Returns the log-likelihood and degrees of freedom for the nested-dichotomies model. (and through it `AIC` and `BIC` model-comparison statistics).

Usage

```
## S3 method for class 'nestedLogit'
Anova(mod, ...)

## S3 method for class 'Anova.nestedLogit'
print(x, ...)

## S3 method for class 'nestedLogit'
linearHypothesis(model, ...)

## S3 method for class 'nestedLogit'
anova(object, object2, ...)

## S3 method for class 'anova.nestedLogit'
print(x, ...)

## S3 method for class 'nestedLogit'
logLik(object, ...)
```

Arguments

`...`	arguments to be passed down. In the case of `linearHypothesis`, the second argument is typically the `hypothesis.matrix`. See the Details section of `linearHypothesis`. In the case of `anova`, additional sequential `"nestedLogit"` models.
`x`, `object`, `object2`, `mod`, `model`	in most cases, an object of class `"nestedLogit"`.

Value

- The Anova and anova methods return objects of class "Anova.nestedLogit" and "anova.nestedLogit", respectively, each of which contains a list of "anova" objects (see anova) and is usually printed.
- The linearHypothesis method is called for its side effect, printing the result of linear hypothesis tests, and invisibly returns NULL.
- The logLik method returns an object of class "logLik" (see logLik).

Author(s)

John Fox

See Also

Anova, anova, linearHypothesis, logLik, AIC, BIC

Examples

```
# define continuation dichotomies for level of education
cont.dichots <- continuationLogits(c("l.t.highschool",
                                     "highschool",
                                     "college",
                                     "graduate"))
# fit a nested model for the GSS data examining education degree in relation to parent & year
m <- nestedLogit(degree ~ parentdeg + year,
                 cont.dichots,
                 data=GSS)

# Anova and anova tests
car::Anova(m) # type-II (partial) tests

anova(update(m, . ~ . - year), m) # model comparison

# Wald test
car::linearHypothesis(m, c("parentdeghighschool", "parentdegcollege",
                           "parentdeggraduate"))

# log-liklihood, AIC, and BIC
logLik(m)
AIC(m)
BIC(m)
```

---

nestedLogit *Binary Logit Models for Nested Dichotomies*

Description

Fit a related set of binary logit models via the glm function to nested dichotomies, comprising a model for the polytomy. A polytomous response with m categories can be analyzed using $m-1$ binary logit comparisons. When these comparisons are nested, the $m-1$ sub-models are statistically independent. Therefore, the likelihood chi-square statistics for the sub-models are additive and give overall tests for a model for the polytomy. This method was introduced by Fienberg (1980),and subsequently illustrated by Fox(2016) and Friendly & Meyer (2016).

`dichotomy` and `logits` are helper functions to construct the dichotomies.

`continuationLogits` constructs a set of $m-1$ logit comparisons, called continuation logits, for an ordered response. With $m = 4$ levels, say, `A, B, C, D`, considered low to high: The first contrasts `B, C, D` against `A`. The second ignores `A` and contrasts `C, D` against `B`. The second ignores `A, B` and contrasts `D` against `C`.

Usage

```
nestedLogit(formula, dichotomies, data, subset = NULL, contrasts = NULL, ...)

logits(...)

dichotomy(...)

continuationLogits(levels, names, prefix = "above_")
```

Arguments

`formula`	a model formula with the polytomous response on the left-hand side and the usual linear-model-like specification on the right-hand side.
`dichotomies`	specification of the logits for the nested dichotomies, constructed by the `logits` and `dichotomy` functions, or `continuationLogits`. Alternatively, the `dichotomies` can be specified as a nested (i.e., recursive) list, the elements of which can be given optional names. See Details.
`data`	a data frame with the data for the model; unlike in most statistical modeling functions, the `data` argument is required. Cases with `NA`s in any of the variables appearing in the model formula will be removed with a Note message.
`subset`	a character string specifying an expression to fit the model to a subset of the data; the default, `NULL`, uses the full data set.
`contrasts`	an optional list of contrast specification for specific factors in the model; see `lm` for details.
`...`	for `nestedLogit`, optional named arguments to be passed to `glm`; for `logits`, definitions of the nested logits—with each named argument specifying a dichotomy; for `dichotomy`, two character vectors giving the levels defining the dichotomy; the vectors can optionally be named.
`levels`	A character vector of set of levels of the variables or a number specifying the numbers of levels (in which case, uppercase letters will be use for the levels).
`names`	Names to be assigned to the dichotomies; if absent, names will be generated from the levels.

prefix — a character string (default: "above_") used as a prefix to the names of the continuation dichotomies.

Details

A *dichotomy* for a categorical variable is a comparison of one subset of levels against another subset. A set of dichotomies is *nested*, if after an initial dichotomy, all subsequent ones are *within* the groups of levels lumped together in earlier ones. Nested dichotomies correspond to a binary tree of the successive divisions.

For example, for a 3-level response, a first dichotomy could be {A}, {B, C} and then the second one would be just {B}, {C}. Note that in the second dichotomy, observations with response A are treated as NA.

The function dichotomy constructs a *single* dichotomy in the required form, which is a list of length 2 containing two character vectors giving the levels defining the dichotomy. The function logits is used to create the set of dichotomies for a response factor. Alternatively, the nested dichotomies can be specified more compactly as a nested (i.e., recursive) list with optionally named elements; for example, list(air="plane", ground=list(public=list("train", "bus"), private="car")).

The function continuationLogits provides a convenient way to generate all dichotomies for an ordered response. For an ordered response with $m = 4$ levels, say, A, B, C, D, considered low to high: The dichotomy first contrasts B, C, D against A. The second ignores A and contrasts C, D against B. The second ignores A, B and contrasts D against C.

Value

nestedLogit returns an object of class "nestedLogit" containing the following elements:

- models, a named list of (normally) $m - 1$ "glm" objects, each a binary logit model for one of the $m - 1$ nested dichotomies representing the m-level response.
- formula, the model formula for the nested logit models.
- dichotomies, the "dichotomies" object defining the nested dichotomies for the model.
- data.name, the name of the data set to which the model is fit, of class "name".
- data, the data set to which the model is fit.
- subset, a character representation of the subset argument or "NULL" if the argument isn't specified.
- contrasts, the contrasts argument or NULL if the argument isn't specified.
- contrasts.print a character representation of the contrasts argument or "NULL" if the argument isn't specified.

logits and continuationLogits return objects of class "dichotomies" and c("continuationDichotomies" "dichotomies"), respectively, which are two-elements lists, each element containing a list of two character vectors representing a dichotomy. dichotomy returns a list of two character vectors representing a dichotomy.

Author(s)

John Fox

References

S. Fienberg (1980). *The Analysis of Cross-Classified Categorical Data*, 2nd Edition, MIT Press, Section 6.6.

J. Fox (2016), *Applied Linear Regression and Generalized Linear Models*, 3rd Edition, Sage, Section 14.2.2.

J. Fox and S. Weisberg (2011), *An R Companion to Applied Regression*, 2nd Edition, Sage, Section 5.8.

M. Friendly and D. Meyers (2016), *Discrete Data Analysis with R*, CRC Press, Section 8.2.

See Also

nestedMethods

Examples

```
data("Womenlf", package = "carData")

  #' Use `logits()` and `dichotomy()` to specify the comparisons of interest
  comparisons <- logits(work=dichotomy("not.work",
                                       working=c("parttime", "fulltime")),
                        full=dichotomy("parttime", "fulltime"))
  print(comparisons)

  m <- nestedLogit(partic ~ hincome + children,
                   dichotomies = comparisons,
                   data=Womenlf)
  print(summary(m))
  print(car::Anova(m))
  coef(m)

  # equivalent;
  nestedLogit(partic ~ hincome + children,
              dichotomies = list("not.work",
                                 working=list("parttime", "fulltime")),
              data=Womenlf)

  # get predicted values
  new <- expand.grid(hincome=seq(0, 45, length=10),
                     children=c("absent", "present"))
  pred.nested <- predict(m, new)

  # plot
  op <- par(mfcol=c(1, 2), mar=c(4, 4, 3, 1) + 0.1)
  plot(m, "hincome", list(children="absent"),
       xlab="Husband's Income", legend=FALSE)
  plot(m, "hincome", list(children="present"),
       xlab="Husband's Income")
  par(op)
```

```
continuationLogits(c("none", "gradeschool", "highschool", "college"))
continuationLogits(4)
```

---

nestedMethods *Methods for* "nestedLogit" *and Related Objects*

Description

Various methods for processing "nestedLogit" and related objects. Most of these are the standard methods for a model-fitting function.

coef, vcov Return the coefficients and their variance-covariance matrix respectively.

update Re-fit a "nestedLogit" model with a change in any of the formula, dichotomies, data, subset, or contrasts, arguments.

predict, fitted Computes predicted values from a fitted "nestedLogit" model.

confint Compute point-wise confidence limits for predicted response-category probabilities or logits.

glance Construct a single row summaries for the dichotomies "nestedLogit" model.

tidy Summarizes the terms in "nestedLogit" model.

Usage

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

## S3 method for class 'nestedLogit'
summary(object, ...)

## S3 method for class 'summary.nestedLogit'
print(x, ...)

## S3 method for class 'dichotomies'
print(x, ...)

## S3 method for class 'nestedLogit'
predict(object, newdata, model = c("nested", "dichotomies"), ...)

## S3 method for class 'predictNestedLogit'
print(x, n = min(10L, nrow(x$p)), ...)

## S3 method for class 'predictNestedLogit'
confint(
  object,
  parm = c("prob", "logit"),
```

```
  level = 0.95,
  conf.limits.logit = TRUE,
  ...
)

## S3 method for class 'predictDichotomies'
print(x, n = 10L, ...)

## S3 method for class 'nestedLogit'
fitted(object, model = c("nested", "dichotomies"), ...)

## S3 method for class 'nestedLogit'
coef(object, as.matrix = TRUE, ...)

## S3 method for class 'nestedLogit'
vcov(object, as.matrix = FALSE, ...)

## S3 method for class 'nestedLogit'
update(object, formula, dichotomies, data, subset, contrasts, ...)

## S3 method for class 'dichotomies'
as.matrix(x, ...)

## S3 method for class 'dichotomies'
as.character(x, ...)

## S3 method for class 'continuationDichotomies'
as.matrix(x, ...)

as.dichotomies(x, ...)

## S3 method for class 'matrix'
as.dichotomies(x, ...)
```

Arguments

x, object	in most cases, an object of class "nestedLogit".
...	arguments to be passed down.
newdata	For the predict method, a data frame containing combinations of values of the predictors at which fitted probabilities (or other quantities) are to be computed.
model	For the predict and fitted methods, either "nested" (the default), in which case fitted probabilities under the nested logit model are returned, or "dichotomies", in which case predict.glm is invoked for each binary logit model fit to the nested dichotomies and a named list of the results is returned.
n	For the print method of predict.nestedLogit or predictDichotomies, an integer or "all" to control how many rows are printed for each of the probabilities of response categories, corresponding logits and their standard errors.

parm	For the confint method, one of "prob" or "logit", indicating whether to generate confidence intervals for probabilities or logits of the responses.
level	Confidence level for the confint method
conf.limits.logit	When parm = "prob" ?????
as.matrix	if TRUE (the default for coef) return coefficients as a matrix with one column for each nested dichotomy, or coefficient covariances as a matrix with one row and column for each combination of dichotomies and coefficients; if FALSE (the default for vcov), return a list of coefficients or coefficient covariances with one element for each dichotomy.
formula	optional updated model formula.
dichotomies	optional updated dichotomies object.
data	optional updated data argument
subset	optional updated subset argument.
contrasts	optional updated contrasts argument.

Details

The predict method provides predicted values for two representations of the model. model = "nested" gives the fitted probabilities for each of the response categories. model = "dichotomies" gives the fitted log odds for each binary logit models in the dichotomies.

Value

- The coef and vcov methods return either matrices or lists of regression coefficients and their covariances, respectively.
- The update method returns an object of class "nestedLogit" (see nestedLogit) derived from the original nested-logit model.
- The predict and fitted methods return an object of class "predictNested" or "predictDichotomies", which contain the predicted probabilities, predicted logits, and other information, such as standard errors of predicted values, and, if supplied, the newdata on which predictions are based.
- The summary method returns an object of class "summary.nestedLogit", which is a list of summaries of the glm objects that comprise the nested-dichotomies model; the object is normally printed.
- The methods for as.matrix, as.character, and as.dichotomies coerce various objects to matrices, character vectors, and dichotomies objects.
- The various print methods invisibly return their x arguments.

Author(s)

John Fox and Michael Friendly

See Also

nestedLogit, plot.nestedLogit, glance.nestedLogit, tidy.nestedLogit

Examples

```
# define continuation dichotomies for level of education
cont.dichots <- continuationLogits(c("l.t.highschool",
                                     "highschool",
                                     "college",
                                     "graduate"))

# Show dichotomies in various forms
print(cont.dichots)
as.matrix(cont.dichots)
as.character(cont.dichots)

# fit a nested model for the GSS data examining education degree in relation to parent & year
m <- nestedLogit(degree ~ parentdeg + year,
                 cont.dichots,
                 data=GSS)

coef(m)                              # coefficient estimates
sqrt(diag(vcov(m, as.matrix=TRUE))) # standard errors
print(m)
summary(m)

# broom methods
broom::glance(m)
broom::tidy(m)

# predicted probabilities and ploting
predict(m) # fitted probabilities for first few cases;

new <- expand.grid(parentdeg=c("l.t.highschool",  "highschool",
                               "college", "graduate"),
                   year=c(1972, 2016))
fit <- predict(m, newdata=new)
cbind(new, fit) # fitted probabilities at specific values of predictors

# predicted logits for dichotomies
predictions <- predict(m, newdata=new, model="dichotomies")
predictions
```

`plot.nestedLogit` *Plotting Nested Logit Models*

Description

A `plot` method for `"nestedLogit"` objects produced by the `nestedLogit` function. Fitted probabilities under the model are plotted for each level of the polytomous response variable, with one of the explanatory variables on the horizontal axis and other explanatory variables fixed to particular values. By default, a 95% pointwise confidence envelope is added to the plot.

Usage

```
## S3 method for class 'nestedLogit'
plot(
  x,
  x.var,
  others,
  n.x.values = 100L,
  xlab = x.var,
  ylab = "Fitted Probability",
  main,
  cex.main = 1,
  digits.main = getOption("digits") - 2L,
  font.main = 1L,
  pch = 1L:length(response.levels),
  lwd = 3,
  lty = 1L:length(response.levels),
  col = palette()[1L:length(response.levels)],
  legend = TRUE,
  legend.inset = 0.01,
  legend.location = "topleft",
  legend.bty = "n",
  conf.level = 0.95,
  conf.alpha = 0.3,
  ...
)
```

Arguments

x	an object of "nestedLogit" produced by nestedLogit.
x.var	quoted name of the variable to appear on the x-axis; if omitted, the first predictor in the model is used.
others	a named list of values for the other variables in the model, that is, other than x.var; if any other predictor is omitted, it is set to an arbitrary value—the mean for a numeric predictor or the first level or value of a factor, character, or logical predictor; only one value may be specified for each variable in others.
n.x.values	the number of evenly spaced values of x.var at which to evaluate fitted probabilities to be plotted (default 100).
xlab	label for the x-axis (defaults to the value of x.var).
ylab	label for the y-axis (defaults to "Fitted Probability").
main	main title for the graph (if missing, constructed from the variables and values in others).
cex.main	size of main title (see par).
digits.main	number of digits to retain when rounding values for the main title.
font.main	font for main title (see par).
pch	plotting characters (see par).

lwd	line width (see par).
lty	line types (see par).
col	line colors (see par).
legend	if TRUE (the default), add a legend for the response levels to the graph.
legend.inset	default 0.01 (see legend).
legend.location	position of the legend (default "topleft", see legend).
legend.bty	the type of box to be drawn around the legend. The allowed values are "o" (the default) and "n".
conf.level	the level for pointwise confidence envelopes around the predicted response probabilities; the default is .0.95. If NULL, the confidence envelopes are suppressed.
conf.alpha	the opacity of the confidence envelopes; the default is 0.3.
...	arguments to be passed to matplot.

Value

NULL Used for its side-effect of producing a plot

Author(s)

John Fox <jfox@mcmaster.ca>

See Also

nestedLogit, matplot

Examples

```
data("Womenlf", package = "carData")
m <- nestedLogit(partic ~ hincome + children,
                 logits(work=dichotomy("not.work", c("parttime", "fulltime")),
                        full=dichotomy("parttime", "fulltime")),
                        data=Womenlf)
plot(m, legend.location="top")
op <- par(mfcol=c(1, 2), mar=c(4, 4, 3, 1) + 0.1)
plot(m, "hincome", list(children="absent"),
     xlab="Husband's Income", legend=FALSE)
plot(m, "hincome", list(children="present"),
     xlab="Husband's Income")
par(op)
```

Index