R topics documented:

cars_china .................................................. 2
cars_us .................................................... 3
dummyCode .................................................. 5
logitr ......................................................... 5
miscmethods.logitr ...................................... 9
predictChoices .............................................. 10
predictProbs ............................................... 12
recodeData .................................................. 13
simulateShares ............................................. 14
statusCodes ................................................ 15
wtp .......................................................... 16
wtpCompare ............................................... 17
yogurt ....................................................... 18

Index 20

cars_china  Stated car choice observations by Chinese car buyers

Description

Data from Helveston et al. (2015) containing 448 stated choice observations from Chinese car
buyers and 384 stated choice observations from US car buyers. Conjoint surveys were fielded in
2012 in four major Chinese cities (Beijing, Shanghai, Shenzhen, and Chengdu), online in the US
on Amazon Mechanical Turk, and in person at the Pittsburgh Auto show. Participants were asked
to select a vehicle from a set of three alternatives. Each participant answered 15 choice questions.

Usage

data(cars_china)

Format

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>individual identifiers</td>
</tr>
<tr>
<td>obsnum</td>
<td>identifier for unique choice observation</td>
</tr>
<tr>
<td>choice</td>
<td>dummy code for choice (1 or 0)</td>
</tr>
<tr>
<td>hev</td>
<td>dummy code for HEV vehicle type (1 or 0)</td>
</tr>
<tr>
<td>phev10</td>
<td>dummy code for PHEV vehicle type w/10 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>phev20</td>
<td>dummy code for PHEV vehicle type w/20 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>phev40</td>
<td>dummy code for PHEV vehicle type w/40 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>bev75</td>
<td>dummy code for BEV vehicle type w/75 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>bev100</td>
<td>dummy code for BEV vehicle type w/100 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>bev150</td>
<td>dummy code for BEV vehicle type w/150 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>phevFastcharge</td>
<td>dummy code for whether PHEV vehicle had fast charging capability (1 or 0)</td>
</tr>
<tr>
<td>bevFastcharge</td>
<td>dummy code for whether BEV vehicle had fast charging capability (1 or 0)</td>
</tr>
</tbody>
</table>
Source
Raw data downloaded from this repo

References

Examples

```r
data(cars_china)
head(cars_china)
```

```r
cars_us
```

*Stated car choice observations by US car buyers*

Description
Data from Helveston et al. (2015) containing 448 stated choice observations from Chinese car buyers and 384 stated choice observations from US car buyers. Conjoint surveys were fielded in 2012 in four major Chinese cities (Beijing, Shanghai, Shenzhen, and Chengdu), online in the US on Amazon Mechanical Turk, and in person at the Pittsburgh Auto show. Participants were asked to select a vehicle from a set of three alternatives. Each participant answered 15 choice questions.

Usage

```r
data(cars_us)
```
Format

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>individual identifiers</td>
</tr>
<tr>
<td>obsnum</td>
<td>identifier for unique choice observation</td>
</tr>
<tr>
<td>choice</td>
<td>dummy code for choice (1 or 0)</td>
</tr>
<tr>
<td>hev</td>
<td>dummy code for HEV vehicle type (1 or 0)</td>
</tr>
<tr>
<td>phev10</td>
<td>dummy code for PHEV vehicle type w/10 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>phev20</td>
<td>dummy code for PHEV vehicle type w/20 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>phev40</td>
<td>dummy code for PHEV vehicle type w/40 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>bev75</td>
<td>dummy code for BEV vehicle type w/75 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>bev100</td>
<td>dummy code for BEV vehicle type w/100 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>bev150</td>
<td>dummy code for BEV vehicle type w/150 mile electric driving range (1 or 0)</td>
</tr>
<tr>
<td>phevFastcharge</td>
<td>dummy code for whether PHEV vehicle had fast charging capability (1 or 0)</td>
</tr>
<tr>
<td>bevFastcharge</td>
<td>dummy code for whether BEV vehicle had fast charging capability (1 or 0)</td>
</tr>
<tr>
<td>price</td>
<td>price of vehicle ($USD)</td>
</tr>
<tr>
<td>opCost</td>
<td>operating cost of vehicle (US cents / mile)</td>
</tr>
<tr>
<td>accelTime</td>
<td>0-60 mph acceleration time (seconds)</td>
</tr>
<tr>
<td>american</td>
<td>dummy code for whether American brand (1 or 0)</td>
</tr>
<tr>
<td>japanese</td>
<td>dummy code for whether Japanese brand (1 or 0)</td>
</tr>
<tr>
<td>chinese</td>
<td>dummy code for whether Chinese brand (1 or 0)</td>
</tr>
<tr>
<td>skorean</td>
<td>dummy code for whether S. Korean brand (1 or 0)</td>
</tr>
<tr>
<td>weights</td>
<td>weights for each individual computed so that the sample age and income demographics matched with those of the original population</td>
</tr>
</tbody>
</table>

Source

Raw data downloaded from this repo

References


Examples

data(cars_us)

head(cars_us)
**dummyCode**

Add dummy-coded variables to data frame.

**Description**

This function is deprecated. Use fastDummies::dummy_cols() instead.

**Usage**

dummyCode(df, vars)

**Arguments**

- `df`: A data frame.
- `vars`: The variables in the data frame for which you want to create new dummy coded variables.

**Value**

A data frame with new dummy-coded variables added.

---

**logitr**

The main function for estimating logit models

**Description**

Use this function to estimate multinomial (MNL) and mixed logit (MXL) models with "Preference" space or "Willingness-to-pay" (WTP) space utility parameterizations. The function includes an option to run a multistart optimization loop with random starting points in each iteration, which is useful for non-convex problems like MXL models or models with WTP space utility parameterizations. The main optimization loop uses the `nloptr()` function to minimize the negative log-likelihood function.

**Usage**

logitr(
  data,  
  choice,  
  obsID,  
  pars,  
  price = NULL,  
  randPars = NULL,  
  randPrice = NULL,  
  modelSpace = "pref",  
  weights = NULL,  
  ...  
)
logitr

panelID = NULL,
clusterID = NULL,
robust = FALSE,
umMultiStarts = 1,
useAnalyticGrad = TRUE,
scaleInputs = TRUE,
startParBounds = c(-1, 1),
standardDraws = NULL,
numDraws = 50,
startVals = NULL,
options = list(print_level = 0, xtol_rel = 1e-06, xtol_abs = 1e-06, ftol_rel = 1e-06,
            ftol_abs = 1e-06, maxeval = 1000, algorithm = "NLOPT_LD_LBFGS"),
parNames,
choiceName,
obsIDName,
priceName,
weightsName,
clusterName,
cluster
)

Arguments

data            The choice data, formatted as a data.frame object.
choice          The name of the column that identifies the choice variable.
obID            The name of the column that identifies each choice observation.
pars            The names of the parameters to be estimated in the model. Must be the same as
                the column names in the data argument. For WTP space models, do not include
                price in pars.
price           The name of the column that identifies the price variable. Required for WTP
                space models. Defaults to NULL.
randPars        A named vector whose names are the random parameters and values the distribu-
                tion: 'n' for normal or 'ln' for log-normal. Defaults to NULL.
randPrice       The random distribution for the price parameter: 'n' for normal or 'ln' for
                log-normal. Only used for WTP space MXL models. Defaults to NULL.
modelSpace      Set to 'wtp' for WTP space models. Defaults to "pref".
weights         The name of the column that identifies the weights to be used in model estima-
                tion. Defaults to NULL.
panelID         The name of the column that identifies the individual (for panel data where mul-
                tiple observations are recorded for each individual). Defaults to NULL.
clusterID       The name of the column that identifies the cluster groups to be used in model
                estimation. Defaults to NULL.
robust          Determines whether or not a robust covariance matrix is estimated. Defaults to
                FALSE. Specification of a clusterID or weights will override the user setting
                and set this to 'TRUE' (a warning will be displayed in this case). Replicates the
                functionality of Stata's cmcmmixlogit.
numMultiStarts is the number of times to run the optimization loop, each time starting from a different random starting point for each parameter between startParBounds. Recommended for non-convex models, such as WTP space models and mixed logit models. Defaults to 1.

useAnalyticGrad  
Set to FALSE to use numerically approximated gradients instead of analytic gradients during estimation. For now, using the analytic gradient is faster for MNL models but slower for MXL models. Defaults to TRUE.

cscaleInputs  By default each variable in data is scaled to be between 0 and 1 before running the optimization routine because it usually helps with stability, especially if some of the variables have very large or very small values (e.g. > 10^3 or < 10^-3). Set to FALSE to turn this feature off. Defaults to TRUE.

startParBounds  sets the lower and upper bounds for the starting parameters for each optimization run, which are generated by runif(n,lower,upper). Defaults to c(-1,1).

standardDraws  By default, a new set of standard normal draws are generated during each call to logitr (the same draws are used during each multistart iteration). The user can override those draws by providing a matrix of standard normal draws if desired. Defaults to NULL.

numDraws  The number of Halton draws to use for MXL models for the maximum simulated likelihood. Defaults to 50.

startVals  is vector of values to be used as starting values for the optimization. Only used for the first run if numMultiStarts > 1. Defaults to NULL.

options  A list of options for controlling the nloptr() optimization. Run nloptr::nloptr.print.options() for details.

parNames  No longer used as of v0.2.3 - if provided, this is passed to the pars argument and a warning is displayed.

choiceName  No longer used as of v0.2.3 - if provided, this is passed to the choice argument and a warning is displayed.

obsIDName  No longer used as of v0.2.3 - if provided, this is passed to the obsID argument and a warning is displayed.

priceName  No longer used as of v0.2.3 - if provided, this is passed to the price argument and a warning is displayed.

weightsName  No longer used as of v0.2.3 - if provided, this is passed to the weights argument and a warning is displayed.

clusterName  No longer used as of v0.2.3 - if provided, this is passed to the clusterID argument and a warning is displayed.

cluster  No longer used as of v0.2.3 - if provided, this is passed to the clusterID argument and a warning is displayed.

Details  
The the options argument is used to control the detailed behavior of the optimization and must be passed as a list, e.g. options = list(...). Below are a list of the default options, but other options can be included. Run nloptr::nloptr.print.options() for more details.
Argument Description Default

xtol_rel The relative x tolerance for the nloptr optimization loop. 1.0e-6
xtol_abs The absolute x tolerance for the nloptr optimization loop. 1.0e-6
ftol_rel The relative f tolerance for the nloptr optimization loop. 1.0e-6
ftol_abs The absolute f tolerance for the nloptr optimization loop. 1.0e-6
maxeval The maximum number of function evaluations for the nloptr optimization loop. 1000
algorithm The optimization algorithm that nloptr uses. "NLOPT_LD_LBFGS"
print_level The print level of the nloptr optimization loop. 0

Value

The function returns a list object containing the following objects.

Value Description

coef The model coefficients at convergence.
logLik The log-likelihood value at convergence.
nullLogLik The null log-likelihood value (if all coefficients are 0).
gradient The gradient of the log-likelihood at convergence.
hessian The hessian of the log-likelihood at convergence.
startPars The starting values used.
multistartNumber The multistart run number for this model.
multistartSummary A summary of the log-likelihood values for each multistart run (if more than one multistart was used).
time The user, system, and elapsed time to run the optimization.
iterations The number of iterations until convergence.
message A more informative message with the status of the optimization result.
status An integer value with the status of the optimization (positive values are successes). Use statusCodes()
call The matched call to logitr()
inputs A list of the original inputs to logitr()
data A list of the original data provided to logitr() broken up into components used during model estimation.
numObs The number of observations.
numParams The number of model parameters.
freq The frequency counts of each choice alternative.
modelType The model type, 'mnl' for multinomial logit or 'mxl' for mixed logit.
weightsUsed TRUE or FALSE for whether weights were used in the model.
numClusters The number of clusters.
parSetup A summary of the distributional assumptions on each model parameter ("f"="fixed", "n"="normal distribution", "ln"="log-normal distribution").
parIDs A list identifying the indices of each parameter in coef by a variety of types.
scaleFactors A vector of the scaling factors used to scale each coefficient during estimation.
standardDraws The draws used during maximum simulated likelihood (for MXL models).
options A list of options for controlling the nloptr() optimization. Run nloptr::nloptr.print.options() for details.

Examples

# For more detailed examples, visit
# https://jhelvy.github.io/logitr/articles/

library(logitr)
# Estimate a MNL model in the Preference space
mnl_pref <- logitr(
  data = yogurt,
  choice = "choice",
  obsID = "obsID",
  pars = c("price", "feat", "brand")
)

# Estimate a MNL model in the WTP space, using a 10-run multistart
mnl_wtp <- logitr(
  data = yogurt,
  choice = "choice",
  obsID = "obsID",
  pars = c("feat", "brand"),
  price = "price",
  modelSpace = "wtp",
  numMultiStarts = 10
)

# Estimate a MXL model in the Preference space with "feat" and "brand"
# following normal distributions
mnl_pref <- logitr(
  data = yogurt,
  choice = "choice",
  obsID = "obsID",
  pars = c("price", "feat", "brand"),
  randPars = c(feat = "n", brand = "n")
)

---

**miscmethods.logitr**

**Methods for logitr objects**

**Description**

Miscellaneous methods for logitr class objects.

**Usage**

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

## S3 method for class 'logitr'
terms(x, ...)

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

## S3 method for class 'summary.logitr'
```


predictChoices

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

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

## S3 method for class 'logitr'
print(
    x,
    digits = max(3, getOption("digits") - 2),
    width = getOption("width"),
    ...
)

## S3 method for class 'summary.logitr'
print(
    x,
    digits = max(3, getOption("digits") - 2),
    width = getOption("width"),
    ...
)

## S3 method for class 'logitr'
vcov(object, ...)

## S3 method for class 'logitr_wtp'
print(
    x,
    digits = max(3, getOption("digits") - 2),
    width = getOption("width"),
    ...
)

### Arguments

- `object` is an object of class logitr.
- `...` further arguments.
- `x` is an object of class logitr.
- `digits` the number of digits for printing, defaults to 3.
- `width` the width of the printing.

### predictChoices

**Predict choices**

Returns the expected choices for a set of one or more alternatives based on the results from an estimated model.
predictChoices

Usage

predictChoices(model, alts, altID, obsID = NULL)

Arguments

model The output of a model estimated model using the logitr() function. Include if you want to compare true choices from actual observations (e.g. hold outs) to the predicted choices.

alts A data frame of a set of alternatives for which to predict choices. Each row is an alternative and each column an attribute corresponding to parameter names in the estimated model.

altID The name of the column that identifies each alternative in each set of alternatives.

obsID The name of the column that identifies each set of alternatives. Required if predicting results for more than one set of alternatives. Defaults to NULL (for a single set of alternatives).

Value

A data frame with the predicted choices for each alternative in alts.

Examples

library(logitr)

# Estimate a preference space model
mnl_pref <- logitr(
  data = yogurt,
  choice = "choice",
  obsID = "obsID",
  pars = c("price", "feat", "brand"
)

# You can predict choices for any set of alternative, such as hold out
# samples or within-sample. For this example, choices will be predicted for
# the full yogurt data set, which was used to estimate the model.

# Predict choices using the estimated preference space MNL model
choices <- predictChoices(
  model = mnl_pref,
  alts = yogurt,
  altID = "alt",
  obsID = "obsID"
)

head(choices)

# Compute the accuracy
chosen <- subset(choices, choice == 1)
chosen$correct <- chosen$choice == chosen$choice_predict
predictProbs

\[
\text{sum(chosen$\text{correct}) / nrow(chosen)} \# \% \text{correctly predicted}
\]

### predictProbs

*Predict expected choice probabilities*

**Description**

Returns the expected choice probabilities for a single set or multiple sets of alternatives based on the results from an estimated model.

**Usage**

```r
predictProbs(
  model,
  alts,
  altID,
  obsID = NULL,
  computeCI = TRUE,
  ci = 0.95,
  numDraws = 10^4,
  alpha
)
```

**Arguments**

- `model`  
The output of a model estimated model using the `logitr()` function.
- `alts`  
A data frame of a set of alternatives for which to predict choice probabilities. Each row is an alternative and each column an attribute corresponding to parameter names in the estimated model.
- `altID`  
The name of the column that identifies each alternative in each set of alternatives.
- `obsID`  
The name of the column that identifies each set of alternatives. Required if predicting results for more than one set of alternatives. Defaults to `NULL` (for a single set of alternatives).
- `computeCI`  
Should a confidence interval be computed? Defaults to `TRUE`.
- `ci`  
The sensitivity of the computed confidence interval (CI). Defaults to `ci = 0.95`, reflecting a 95% CI.
- `numDraws`  
The number of draws to use in simulating uncertainty for the computed confidence interval.
- `alpha`  
The sensitivity of the computed confidence interval. No longer used as of v0.2.7 - if provided, a warning is shown and `ci` is computed from `alpha`.

**Value**

A data frame with the estimated choice probabilities for each alternative in `alts`.
Examples

library(logitr)

# Estimate a preference space model
mnl_pref <- logitr(
  data = yogurt,
  choice = "choice",
  obsID = "obsID",
  pars = c("price", "feat", "brand")
)

# Create a set of alternatives for which to predict choice probabilities.
# Each row is an alternative and each column an attribute. In this example,
# two of the choice observations from the yogurt dataset are used
alts <- subset(
  yogurt, obsID %in% c(42, 13),
  select = c('obsID', 'alt', 'price', 'feat', 'brand'))

alts

# Predict choice probabilities using the estimated preference space MNL
# model
predictProbs(mnl_pref, alts, altID = "alt", obsID = "obsID")

recodeData

Returns a list of the design matrix X and updated pars and randPars
to include any dummy-coded categorical or interaction variables.

Description

Recodes the data and returns a list of the encoded design matrix (X) as well as two vectors (pars and randPars) with discrete (categorical) variables and interaction variables added to X, pars, and randPars.

Usage

recodeData(data, pars, randPars)

Arguments

data The choice data, formatted as a data.frame object.
pars The names of the parameters to be estimated in the model. Must be the same as the column names in the data argument. For WTP space models, do not include price in pars.
randPars A named vector whose names are the random parameters and values the distribution: ‘n’ for normal or ‘ln’ for log-normal. Defaults to NULL.
simulateShares

Value

A list of the design matrix (X) and two vectors (pars and randPars) with discrete (categorical) variables and interaction variables added.

Examples

```r
library(logitr)

data(yogurt)

# Recode the yogurt data
result <- recodeData(
  data = yogurt,
  pars = c("price", "feat", "brand", "price*brand"),
  randPars = c(feat = "n", brand = "n")
)

result$pars
result$randPars
head(result$X)
```

simulateShares  Simulate expected shares

Description

This function has been depreciated since logitr version 0.1.4. Use predictProbs() instead.

Usage

```r
simulateShares(
  model,
  alts,
  obsIDName = NULL,
  priceName = NULL,
  computeCI = TRUE,
  alpha = 0.025,
  numDraws = 10^4
)
```

Arguments

- **model**: The output of a model estimated model using the logitr() function.
- **alts**: A data frame of a set of alternatives for which to simulate shares. Each row is an alternative and each column an attribute corresponding to parameter names in the estimated model.
statusCodes

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obsIDName</td>
<td>The name of the column that identifies each set of alternatives. Required if simulating results for more than one set of alternatives. Defaults to NULL (for a single set of alternatives).</td>
</tr>
<tr>
<td>priceName</td>
<td>The name of the parameter that identifies price. Only required for WTP space models. Defaults to NULL.</td>
</tr>
<tr>
<td>computeCI</td>
<td>Should a confidence interval be computed? Defaults to TRUE.</td>
</tr>
<tr>
<td>alpha</td>
<td>The sensitivity of the computed confidence interval. Defaults to alpha = 0.025, reflecting a 95% CI.</td>
</tr>
<tr>
<td>numDraws</td>
<td>The number of draws to use in simulating uncertainty for the computed confidence interval.</td>
</tr>
</tbody>
</table>

Value

A data frame with the estimated shares for each alternative in alts.

statusCodes()  

Description

Prints a description of the status codes from the nloptr optimization routine.

Usage

statusCodes()

Value

No return value; prints a summary of the nloptr status codes to the console.

Examples

statusCodes()
Get WTP from a preference space model

Description

Returns the computed WTP from a preference space model.

Usage

wtp(model, price)

Arguments

model The output of a "preference space" model estimated using the logitr() function.
price The name of the parameter that identifies price.

Details

Willingness to pay is computed by dividing the estimated parameters of a utility model in the "preference" space by the price parameter. Uncertainty is handled via simulation.

Value

A data frame of the WTP estimates.

Examples

library(logitr)

# Estimate a preference space model
mnl_pref <- logitr(
    data = yogurt,
    choice = "choice",
    obsID = "obsID",
    pars = c("price", "feat", "brand")
)

# Compute the WTP implied from the preference space model
wtp(mnl_pref, price = "price")
Description

Returns a comparison of the WTP between a preference space and WTP space model.

Usage

wtpCompare(model_pref, model_wtp, price)

Arguments

model_pref The output of a "preference space" model estimated using the logitr() function.
model_wtp The output of a "willingness to pay space" model estimated using the logitr() function.
price The name of the parameter that identifies price.

Details

Willingness to pay (WTP) is first computed from the preference space model by dividing the estimated parameters by the price parameter. Then those estimates are compared against the WTP values directly estimated from the "WTP" space model. Uncertainty is handled via simulation.

Value

A data frame comparing the WTP estimates from preference space and WTP space models.

Examples

library(logitr)

# Estimate a MNL model in the Preference space
mnl_pref <- logitr(
    data = yogurt,
    choice = "choice",
    obsID = "obsID",
    pars = c("price", "feat", "brand")
)

# Compute the WTP implied from the preference space model
wtp_mnl_pref <- wtp(mnl_pref, price = "price")

# Estimate a MNL model in the WTP Space, using the computed WTP values
# from the preference space model as starting points
mnl_wtp <- logitr(
    data = yogurt,
choice = "choice",
obsID = "obsID",
pars = c("feat", "brand"),
price = "price",
modelSpace = "wtp",
startVals = wtp_mnl_pref$Estimate
)

# Compare the WTP between the two spaces
wtpCompare(mnl_pref, mnl_wtp, price = "price")

yogurt

Choice observations of yogurt purchases by 100 households

Description

Data from Jain et al. (1994) containing 2,412 choice observations from a series of yogurt purchases by a panel of 100 households in Springfield, Missouri, over a roughly two-year period. The data were collected by optical scanners and contain information about the price, brand, and a "feature" variable, which identifies whether a newspaper advertisement was shown to the customer. There are four brands of yogurt: Yoplait, Dannon, Weight Watchers, and Hiland, with market shares of 34%, 40%, 23% and 3%, respectively.

Usage

data(yogurt)

Format

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>individual identifiers</td>
</tr>
<tr>
<td>obsID</td>
<td>identifier for unique choice observation</td>
</tr>
<tr>
<td>alt</td>
<td>alternative in each choice observation</td>
</tr>
<tr>
<td>choice</td>
<td>dummy code for choice (1 or 0)</td>
</tr>
<tr>
<td>price</td>
<td>price of yogurt</td>
</tr>
<tr>
<td>feat</td>
<td>dummy for whether a newspaper advertisement was shown to the customer (1 or 0)</td>
</tr>
<tr>
<td>brand</td>
<td>yogurt brand: &quot;yoplait&quot;, &quot;dannon&quot;, &quot;hiland&quot;, or &quot;weight&quot; (for weight watcher)</td>
</tr>
<tr>
<td>dannon</td>
<td>dummy variable for the &quot;dannon&quot; brand (1 or 0)</td>
</tr>
<tr>
<td>hiland</td>
<td>dummy variable for the &quot;hiland&quot; brand (1 or 0)</td>
</tr>
<tr>
<td>weight</td>
<td>dummy variable for the &quot;weight&quot; brand (1 or 0)</td>
</tr>
<tr>
<td>yoplait</td>
<td>dummy variable for the &quot;yoplait&quot; brand (1 or 0)</td>
</tr>
</tbody>
</table>
Source
Raw data downloaded from the package mlogit v0.3-0 by Yves Croissant archive

References

Examples

```r
data(yogurt)
head(yogurt)
```
Index

* choice
  predictChoices, 10
* codes
  statusCodes, 15
* datasets
  cars_china, 2
cars_us, 3
yogurt, 18
* logitr
  logitr, 5
  predictChoices, 10
  predictProbs, 12
  simulateShares, 14
  statusCodes, 15
  wtp, 16
  wtpCompare, 17
* logit
  logitr, 5
* mixed
  logitr, 5
* mnl
  logitr, 5
* mxl
  logitr, 5
* nlopter
  statusCodes, 15
* predict
  predictChoices, 10
  predictProbs, 12
* probabilities
  predictProbs, 12
* simulation
  predictChoices, 10
  predictProbs, 12
  simulateShares, 14
* status
  statusCodes, 15
* willingness-to-pay
  logitr, 5

* wtp
  logitr, 5
  wtp, 16
  wtpCompare, 17
cars_china, 2
cars_us, 3
coef.logitr (miscmethods.logitr), 9
coef.summary.logitr
  (miscmethods.logitr), 9
dummyCode, 5
logitr, 5
logLik.logitr (miscmethods.logitr), 9
miscmethods.logitr, 9
predictChoices, 10
predictProbs, 12
print.logitr (miscmethods.logitr), 9
print.logitr_wtp (miscmethods.logitr), 9
print.summary.logitr
  (miscmethods.logitr), 9
recodeData, 13
simulateShares, 14
statusCodes, 15
statusCodes(), 8
summary.logitr (miscmethods.logitr), 9
terms.logitr (miscmethods.logitr), 9
vcov.logitr (miscmethods.logitr), 9
wtp, 16
wtpCompare, 17
yogurt, 18