Package ‘logitr’

January 4, 2022

Title Logit Models w/Preference & WTP Space Utility Parameterizations

Version 0.5.0

Description Fast estimation of multinomial (MNL) and mixed logit (MXL) models in R. Models can be estimated using ``Preference'' space or ``Willingness-to-pay'' (WTP) space utility parameterizations. Weighted models can also be estimated. An option is available to run a parallelized 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' package to minimize the negative log-likelihood function. Additional functions are available for computing and comparing WTP from both preference space and WTP space models and for predicting expected choices and choice probabilities for sets of alternatives based on an estimated model. MXL models assume uncorrelated heterogeneity covariances and are estimated using maximum simulated likelihood based on the algorithms in Train (2009) ``Discrete Choice Methods with Simulation, 2nd Edition'' <doi:10.1017/CBO9780511805271>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

VignetteBuilder knitr

Depends R (>= 3.5.0)

Suggests dplyr, fastDummies, knitr, rmarkdown, here, ggplot2, testthat

Imports nloptr, parallel, stats, randtoolbox, MASS

URL `https://github.com/jhelvy/logitr`

BugReports `https://github.com/jhelvy/logitr/issues`

NeedsCompilation no

Author John Helveston [aut, cre, cph]
(<`https://orcid.org/0000-0002-2657-9191`>),
Connor Forsythe [ctb]

Maintainer John Helveston <`john.helveston@gmail.com`>

Repository CRAN

Date/Publication 2022-01-04 21:40:06 UTC

R topics documented:

`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

Variable	Description
`id`	individual identifiers
`obsnum`	identifier for unique choice observation
`choice`	dummy code for choice (`1` or `0`)
`hev`	dummy code for HEV vehicle type (`1` or `0`)
`phev10`	dummy code for PHEV vehicle type w/10 mile electric driving range (`1` or `0`)
`phev20`	dummy code for PHEV vehicle type w/20 mile electric driving range (`1` or `0`)
`phev40`	dummy code for PHEV vehicle type w/40 mile electric driving range (`1` or `0`)
`bev75`	dummy code for BEV vehicle type w/75 mile electric driving range (`1` or `0`)

`bev100`	dummy code for BEV vehicle type w/100 mile electric driving range (1 or 0)
`bev150`	dummy code for BEV vehicle type w/150 mile electric driving range (1 or 0)
`phevFastcharge`	dummy code for whether PHEV vehicle had fast charging capability (1 or 0)
`bevFastcharge`	dummy code for whether BEV vehicle had fast charging capability (1 or 0)
`price`	price of vehicle ($USD)
`opCost`	operating cost of vehicle (US cents / mile)
`accelTime`	0-60 mph acceleration time (seconds)
`american`	dummy code for whether American brand (1 or 0)
`japanese`	dummy code for whether Japanese brand (1 or 0)
`chinese`	dummy code for whether Chinese brand (1 or 0)
`skorean`	dummy code for whether S. Korean brand (1 or 0)
`weights`	weights for each individual computed so that the sample age and income demographics matched with thos

Source

Raw data downloaded from this repo

References

Helveston, J. P., Liu, Y., Feit, E. M., Fuchs, E. R. H., Klampfl, E., & Michalek, J. J. (2015). "Will Subsidies Drive Electric Vehicle Adoption? Measuring Consumer Preferences in the U.S. and China." Transportation Research Part A: Policy and Practice, 73, 96–112. doi: 10.1016/j.tra.2015.01.002

Examples

```
data(cars_china)

head(cars_china)
```

`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

```
data(cars_us)
```

Format

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

Source

Raw data downloaded from this repo

References

Helveston, J. P., Liu, Y., Feit, E. M., Fuchs, E. R. H., Klampfl, E., & Michalek, J. J. (2015). "Will Subsidies Drive Electric Vehicle Adoption? Measuring Consumer Preferences in the U.S. and China." Transportation Research Part A: Policy and Practice, 73, 96–112. doi: 10.1016/j.tra.2015.01.002

Examples

```
data(cars_us)

head(cars_us)
```

---

fitted.logitr *Extract Model Fitted Values*

Description

Returns fitted values from an object of class logitr.

Usage

```
## S3 method for class 'logitr'
fitted(object, probs = NULL, ...)
```

Arguments

object	is an object of class `logitr` (a model estimated using the 'logitr()' function).
probs	Predicted probabilities for an object of class `logitr` to use in computing fitted values Defaults to `NULL`.
...	further arguments.

Value

A data frame of the `obsID` and the fitted values extracted from `object`.

Examples

```
library(logitr)

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

# Extract the fitted values from the model
fitted(mnl_pref)
```

---

fquantile *Predict probabilities and / or outcomes*

Description

This function is a faster implementation of the "type 7" `quantile()` algorithm and is modified from this gist: https://gist.github.com/sikli/f1775feb9736073cefee97ec81f6b193 It returns sample quantiles corresponding to the given probabilities. The smallest observation corresponds to a probability of 0 and the largest to a probability of 1. For speed, output quantile names are removed as are error handling such as checking if x are factors, or if probs lie outside the [0,1] range.

Usage

```
fquantile(x, probs = seq(0, 1, 0.25), na.rm = FALSE)
```

Arguments

x	numeric vector whose sample quantiles are wanted. `NA` and `NaN` values are not allowed in numeric vectors unless `na.rm` is `TRUE`.
probs	numeric vector of probabilities with values in [0,1]. (Values up to `2e-14` outside that range are accepted and moved to the nearby endpoint.)
na.rm	logical; if `TRUE`, any `NA` and `NaN`'s are removed from `x` before the quantiles are computed.

Value

A vector of length `length(probs)` is returned;

Examples

```
library(logitr)
```

`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,
  outcome,
  obsID,
  pars,
  price = NULL,
  randPars = NULL,
  randPrice = NULL,
  modelSpace = "pref",
  weights = NULL,
  panelID = NULL,
  clusterID = NULL,
  robust = FALSE,
  startParBounds = c(-1, 1),
  startVals = NULL,
```

```
  numMultiStarts = 1,
  useAnalyticGrad = TRUE,
  scaleInputs = TRUE,
  standardDraws = NULL,
  numDraws = 50,
  numCores = NULL,
  vcov = FALSE,
  predict = TRUE,
 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"),
  choice,
  parNames,
  choiceName,
  obsIDName,
  priceName,
  weightsName,
  clusterName,
  cluster
)
```

Arguments

data	The data, formatted as a `data.frame` object.
outcome	The name of the column that identifies the outcome variable, which should be coded with a `1` for `TRUE` and `0` for `FALSE`.
obsID	The name of the column that identifies each 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 distribution: `'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 estimation. Defaults to `NULL`.
panelID	The name of the column that identifies the individual (for panel data where multiple 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.

- `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)`.
- `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`.
- `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`.
- `scaleInputs` 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`.
- `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`.
- `numCores` The number of cores to use for parallel processing of the multistart. Set to `1` to serially run the multistart. Defaults to `NULL`, in which case the number of cores is set to `parallel::detectCores() -1`. Max cores allowed is capped at `parallel::detectCores()`.
- `vcov` Set to `TRUE` to evaluate and include the variance-covariance matrix and coefficient standard errors in the returned object. Defaults to `FALSE`.
- `predict` If `FALSE`, predicted probabilities, fitted values, and residuals are not included in the returned object. Defaults to `TRUE`.
- `options` A list of options for controlling the `nloptr()` optimization. Run `nloptr::nloptr.print.options()` for details.
- `choice` No longer used as of v0.4.0 - if provided, this is passed to the `outcome` argument and a warning is displayed.
- `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 `outcome` 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
`coefficients`	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.
`probabilities`	Predicted probabilities. Not returned if `predict = FALSE`.
`fitted.values`	Fitted values. Not returned if `predict = FALSE`.
`residuals`	Residuals. Not returned if `predict = FALSE`.
`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 estima
`numObs`	The number of observations.
`numParams`	The number of model parameters.
`freq`	The frequency counts of each 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 dis
`parIDs`	A list identifying the indices of each parameter in `coefficients` 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()`

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,
  outcome = "choice",
  obsID   = "obsID",
  pars    = c("price", "feat", "brand")
)

# Estimate a MNL model in the WTP space, using a 5-run multistart
mnl_wtp <- logitr(
  data          = yogurt,
  outcome       = "choice",
  obsID         = "obsID",
  pars          = c("feat", "brand"),
  price         = "price",
  modelSpace    = "wtp",
  numMultiStarts = 5
)

# Estimate a MXL model in the Preference space with "feat"
# following a normal distribution
mxl_pref <- logitr(
  data     = yogurt,
  outcome  = "choice",
  obsID    = "obsID",
  pars     = c("price", "feat", "brand"),
  randPars = c(feat = "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'
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_wtp'
print(
  x,
  digits = max(3, getOption("digits") - 2),
  width = getOption("width"),
  ...
)
```

Arguments

`object`	is an object of class `logitr` (a model estimated using the 'logitr()' function).
`...`	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.

`predict.logitr` *Predict probabilities and / or outcomes*

Description

This method is used for computing predicted probabilities and / or outcomes for either the data used for model estimation or a new data set consisting of a single or multiple sets of alternatives.

Usage

```
## S3 method for class 'logitr'
predict(
  object,
  newdata = NULL,
  obsID = NULL,
  price = NULL,
  type = "prob",
  returnData = FALSE,
  ci = NULL,
  numDrawsCI = 10^3,
  ...
)
```

Arguments

`object`	is an object of class `logitr` (a model estimated using the 'logitr()' function).
`newdata`	a `data.frame`. Each row is an alternative and each column an attribute corresponding to parameter names in the estimated model. Defaults to `NULL`, in which case predictions are made on the original data used to estimate the model.
`obsID`	The name of the column that identifies each set of alternatives in the data. Required if newdata != NULL. Defaults to `NULL`, in which case the value for `obsID` from the data in `object` is used.
`price`	The name of the column that identifies the price variable. Required if the `object` is a WTP space model and if newdata != NULL. Defaults to `NULL`.
`type`	A character vector defining what to predict: `prob` for probabilities, `outcomes` for outcomes. If you want both outputs, use `c("prob","outcome")`. Outcomes are predicted randomly according to the predicted probabilities. Defaults to `"prob"`.
`returnData`	If `TRUE` the data is also returned, otherwise only the predicted values ("prob" and / or "outcome") are returned. Defaults to `FALSE`.
`ci`	If a confidence interval (CI) for the predicted probabilities is desired, set `ci` to a number between 0 and 1 to define the CI sensitivity. For example, `ci = 0.95` will return a 95% CI. Defaults to `NULL`, in which case no CI is computed.
`numDrawsCI`	The number of draws to use in simulating uncertainty for the computed CI. Defaults to 10^3.
`...`	further arguments.

Value

A data frame of predicted probabilities and / or outcomes.

Examples

```
library(logitr)

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

# Predict probabilities and / or outcomes

# Predict probabilities for each alternative in the model data
probs <- predict(mnl_pref)
head(probs)

# Create a set of alternatives for which to make predictions.
# Each row is an alternative and each column an attribute.
data <- subset(
    yogurt, obsID %in% c(42, 13),
    select = c('obsID', 'alt', 'price', 'feat', 'brand'))
data

# Predict probabilities using the estimated model
predict(mnl_pref, newdata = data, obsID = "obsID")

# Predict outcomes
predict(mnl_pref, newdata = data, obsID = "obsID", type = "outcome")

# Predict outcomes and probabilities
predict(mnl_pref, newdata = data, obsID = "obsID", type = c("prob", "outcome"))
```

`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 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`.

Value

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

Examples

```
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)
```

`residuals.logitr` *Extract Model Residuals*

Description

Returns model residuals from an object of class `logitr`.

Usage

```
## S3 method for class 'logitr'
residuals(object, fitted = NULL, ...)
```

Arguments

`object`	is an object of class `logitr` (a model estimated using the 'logitr()' function).
`fitted`	Fitted values for an object of class `logitr` to use in computing residuals. Defaults to `NULL`.
`...`	further arguments.

Value

A data frame of the `obsID` and the residuals (response minus fitted values) extracted from `object`.

Examples

```
library(logitr)

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

# Extract the residuals from the model
residuals(mnl_pref)
```

---

se — *Extract standard errors*

Description

Extract standard errors

Usage

```
se(object, ...)
```

Arguments

`object`	is an object of class `logitr` (a model estimated using the 'logitr()' function).
`...`	further arguments.

---

se.logitr — *Extract standard errors*

Description

Extract standard errors

Usage

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

Arguments

`object` is an object of class `logitr` (a model estimated using the 'logitr()' function).

`...` further arguments.

`statusCodes` *View a description the nloptr status codes*

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()
```

`vcov.logitr` *Calculate the variance-covariance matrix*

Description

Returns the variance-covariance matrix of the main parameters of a fitted model object.

Usage

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

Arguments

`object` is an object of class `logitr` (a model estimated using the 'logitr()' function).

`...` further arguments.

wtp *Get WTP estimates a preference space model*

Description

Returns the computed WTP from a preference space model.

Usage

```
wtp(object, price)
```

Arguments

object	is an object of class `logitr` (a 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,
  outcome = "choice",
  obsID   = "obsID",
  pars    = c("price", "feat", "brand")
)

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

wtp.logitr — *Get WTP from a preference space model*

Description

Returns the computed WTP from a preference space model.

Usage

```
## S3 method for class 'logitr'
wtp(object, price)
```

Arguments

`object`	is an object of class `logitr` (a 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,
  outcome = "choice",
  obsID   = "obsID",
  pars    = c("price", "feat", "brand")
)

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

wtpCompare *Compare WTP from preference and WTP space models*

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,
  outcome = "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,
```

```
  outcome    = "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

Variable	Description
`id`	individual identifiers
`obsID`	identifier for unique choice observation
`alt`	alternative in each choice observation
`choice`	dummy code for choice (1 or 0)
`price`	price of yogurt
`feat`	dummy for whether a newspaper advertisement was shown to the customer (`1` or `0`)
`brand`	yogurt brand: `"yoplait"`, `"dannon"`, `"hiland"`, or `"weight"` (for weight watcher)

Source

Raw data downloaded from the package mlogit v0.3-0 by Yves Croissant archive

References

Dipak C. Jain, Naufel J. Vilcassim & Pradeep K. Chintagunta (1994) A Random-Coefficients Logit Brand-Choice Model Applied to Panel Data, Journal of Business & Economic Statistics, 12:3, 317-328, doi: 10.1080/07350015.1994.10524547

Examples

```
data(yogurt)

head(yogurt)
```

Index