

Package: auctionr (via r-universe)

October 23, 2024

Type Package

Title Estimate First-Price Auction Model

Version 0.1.0

Description Estimates a first-price auction model with conditionally independent private values as described in MacKay (2020) <[doi:10.2139/ssrn.3096534](https://doi.org/10.2139/ssrn.3096534)>. The model allows for unobserved heterogeneity that is common to all bidders in addition to observable heterogeneity.

License GPL-3

URL <https://github.com/ajmack/auctionr>

BugReports <https://github.com/ajmack/auctionr/issues>

Depends R (>= 3.5.0)

Imports stats, parallel, numDeriv (>= 2016.8-1)

VignetteBuilder knitr

Encoding UTF-8

LazyData TRUE

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Suggests testthat (>= 2.1.0), knitr, rmarkdown

Repository <https://ajmack.r-universe.dev>

RemoteUrl <https://github.com/ajmack/auctionr>

RemoteRef HEAD

RemoteSha 03d50a0dc91838f915816fddc713b0f1737b0ae5

Contents

auction_generate_data	2
auction_model	3
print.auctionmodel	5

Index	7
--------------	----------

auction_generate_data *Generates sample data for running [auction_model](#)*

Description

Generates sample data for running [auction_model](#)

Usage

```
auction_generate_data(
  obs = NULL,
  max_n_bids = 10,
  new_x_mean = NULL,
  new_x_sd = NULL,
  mu = NULL,
  alpha = NULL,
  sigma = NULL,
  beta = NULL
)
```

Arguments

obs	Number of observations (or auctions) to draw.
max_n_bids	Maximum number of bids per auction (must be 3 or greater). The routine generates a vector of length obs of random numbers between 2 and max_n_bids.
new_x_mean	Mean values for observable controls to be generated from a Normal distribution.
new_x_sd	Standard deviations for observable controls to be generated from a Normal distribution.
mu	Value for mu, or mean, of private value distribution (Weibull) to be generated.
alpha	Value for alpha, or shape parameter, of private value distribution (Weibull) to be generated.
sigma	Value for standard deviation of unobserved heterogeneity distribution. Note that the distribution is assumed to have mean 1.
beta	Coefficients for the generated observable controls. Must be of the same length as new_x_mean and new_x_sd.

Details

This function generates example data for feeding into [auction_model\(\)](#). Specifically, the winning bid, number of bids, and observed heterogeneity are sampled for the specified number of observations.

Value

A data frame with obs rows and the following columns:

winning_bid	numeric values of the winning bids for each observation
n_bids	number of bids for each observation
X#	X terms that represent observed heterogeneity

See Also

[auction_model](#)

Examples

```
dat <- auction_generate_data(obs = 100,
                             mu = 10,
                             new_x_mean= c(-1,1),
                             new_x_sd = c(0.5,0.8),
                             alpha = 2,
                             sigma = 0.2,
                             beta = c(-1,1))

dim(dat)
head(dat)
```

auction_model	<i>Estimates a first-price auction model.</i>
---------------	---

Description

Estimates a first-price auction model.

Usage

```
auction_model(
  dat = NULL,
  init_param = NULL,
  num_cores = 1,
  method = "BFGS",
  control = list(),
  std_err = FALSE,
  hessian_args = list()
)
```

Arguments

<code>dat</code>	A <code>data.frame</code> containing input columns in the following order: the winning bids, number of bids, and X variables that represent observed heterogeneity.
<code>init_param</code>	Vector of initial values for μ , α , σ , and β vector, provided in order specified. Note that the Weibull distribution requires μ and α to be positive. The standard deviation of unobserved heterogeneity, σ , must be positive as well. The β vector may take any values. If <code>init_params</code> is not provided, all values will be set to 1 by default.
<code>num_cores</code>	The number of cores for running the model in parallel. The default value is 1.
<code>method</code>	Optimization method to be used in <code>optim()</code> (see <code>?optim</code> for details).
<code>control</code>	A list of control parameters to be passed to <code>optim()</code> (see <code>?optim</code> for details).
<code>std_err</code>	If <code>TRUE</code> , the standard errors of the parameters will also be calculated. Note that it may significantly increase the computation time.
<code>hessian_args</code>	A list of arguments passed as the <code>method.args</code> argument of the <code>hessian()</code> function if standard errors are calculated (see <code>?hessian</code> for details).

Details

This function estimates a first-price auction model with conditionally independent private values. This version of the package estimates a procurement auction, where the winning bid is the amount that a single buyer will pay to the top bidding supplier, and values correspond to costs. The model allows for unobserved heterogeneity that is common to all bidders in addition to observable heterogeneity. The winning bid (Y) takes the form

$$Y = B * U * h(X)$$

where B is the proportional winning bid, U is the unobserved heterogeneity, and $h(X)$ controls for observed heterogeneity. The model is log-linear so that $\log(Y) = \log(B) + \log(U) + \log(h(X))$ and $\log(h(X)) = \beta_1 * X_1 + \beta_2 * X_2 + \dots$

The (conditionally) independent private costs are drawn from a Weibull distribution with parameters μ (mean) and α (shape). The CDF of this distribution is given by

$$F(c) = 1 - \exp(- (c * 1/\mu * \text{Gamma}(1 + 1/\alpha))^{\alpha})$$

The unobserved heterogeneity U is sampled from log-normal distribution with mean 1 and a free parameter σ representing its standard deviation.

`init_params`, the initial guess for convergence, must be supplied.

This function utilizes the `Rsnow` framework within the `Rparallel` package. If `numcores` is not specified, this will be run using only one CPU/core. One can use `parallel::detectCores()` to determine how many are available on your system, but you are not advised to use all at once, as this may make your system unresponsive. Please see `Rparallel` and `Rsnow` for more details.

Note that the supplied data can not have missing values.

Value

A list returned by `optim()`. See `?optim` for more details. If `std_err` was set to `TRUE` and the routine succeeded in inverting the estimated Hessian, the list will have an additional component:

<code>std_err</code>	A vector of standard errors for parameter estimates.
----------------------	--

Author(s)

Mackay, Alexander. <amackay@hbs.edu>, HBS Research Computing <research@hbs.edu>.

References

Mackay, Alexander. 2020. "Contract Duration and the Costs of Market Transactions." Working paper, Appendix G.

See Also

[auction_generate_data](#)

Examples

```
#####
## Estimating parameters and standard errors with custom "control" argument
set.seed(100)
dat <- auction_generate_data(obs = 15, mu = 10, alpha = 2,
                             sigma = 0.2, beta = c(-1,1),
                             new_x_mean= c(-1,1),
                             new_x_sd = c(0.5,0.8))

res <- auction_model(dat, init_param = c(8, 2, .5, .4, .6),
                     num_cores = 1,
                     control = list(parscale = c(1,0.1,0.1,1,1)),
                     std_err = TRUE)

res

## run vignette("auctionr") to view a more detailed example
```

print.auctionmodel *Print an auction model.*

Description

Print an auction model.

Usage

```
## S3 method for class 'auctionmodel'
print(x, digits = 6, ...)
```

Arguments

x	An object of class auctionmodel.
digits	Number of digits to display.
...	Additional arguments passed to other methods.

6

print.auctionmodel

Value

x, invisibly.

Index

`auction_generate_data`, 2, 5

`auction_model`, 2, 3, 3

`print.auctionmodel`, 5