Package 'oneinfl'

June 21, 2025

Type Package

Title Estimates OIPP and OIZTNB Regression Models

Version 1.0.2

Author Ryan T. Godwin [aut, cre]

Maintainer Ryan T. Godwin <`ryan.godwin@umanitoba.ca`>

Description Estimates one-inflated positive Poisson (OIPP) and
one-inflated zero-truncated negative binomial (OIZTNB) regression
models. A suite of ancillary statistical tools are also provided,
including: estimation of positive Poisson (PP) and zero-truncated
negative binomial (ZTNB) models; marginal effects and their standard
errors; diagnostic likelihood ratio and Wald tests; plotting;
predicted counts and expected responses; and random variate
generation. The models and tools, as well as four applications, are
shown in Godwin, R. T. (2024). ``One-inflated zero-truncated count
regression models'' arXiv preprint <doi:10.48550/arXiv.2402.02272>.

License GPL (>= 3)

Imports graphics, stats

Encoding UTF-8

NeedsCompilation no

RoxygenNote 7.3.2

Repository CRAN

Date/Publication 2025-06-21 20:20:02 UTC

Contents

makeXZy *Prepare Design Matrices and Response Vector*

Description

Processes a model formula and a data frame to generate design matrices (‘X‘ and ‘Z‘) and a response vector (‘y‘) for regression models, including support for complex formulas with ‘|‘ operators.

Usage

```
makeXZy(formula, df)
```

Arguments

`formula`	A symbolic description of the model, where the left-hand side specifies the response variable and the right-hand side specifies predictors. Formulas can include a ‘|‘ operator to separate predictors for different components of a model.
`df`	A data frame containing the variables specified in the formula.

Details

This function processes the formula to extract and construct: - ‘X‘: The main design matrix. - ‘Z‘: A secondary design matrix (if a ‘|‘ operator is used in the formula, separating components). - ‘y‘: The response variable.

It handles cases where the formula specifies: - Only the main component (e.g., `y ~ x1 + x2`). - A secondary component using the ‘|‘ operator (e.g., `y ~ x1 + x2 | z1 + z2`).

Value

A list containing the following components:

`X` A design matrix for the main predictors.

`Z` A design matrix for additional predictors (e.g., for a secondary process in a two-component model).

`y` The response vector extracted from the formula.

See Also

model.matrix, model.frame, model.response

---

margins *Compute Marginal Effects for One-inflated models*

Description

This wrapper function calls a different function to calculate marginal effects depending on the model type. The marginal effects of the variables are evaluated at specified points, such as the sample means or averages, or at custom-defined cases.

Usage

```
margins(model, df, at = "AE", verbose = TRUE)
```

Arguments

`model`	An object representing a fitted model. Must be of class 'oneinflmodel' or 'truncmodel'.
`df`	A `data.frame` containing the dataset used to fit the model. The variables in the data must match those used in the model.
`at`	A character string or list specifying where to evaluate the marginal effects: • `"AE"`: Average Effect (marginal effect averaged over all data points; default). • `"EM"`: Effect at Means (marginal effect evaluated at the sample means of the data). • A named list: A custom case specifying representative values for variables.
`verbose`	Logical; if `TRUE` (default), prints the summary output. If `FALSE`, suppresses output table and returns a list containing several components.

Details

The function computes marginal effects for zero-truncated Poisson or negative binomial regression models. It handles different model types; 'oneinflmodel' for one-inflated models, and 'truncmodel' for standard count models. The marginal effects are evaluated at either all data points and averaged ('AE', the default), at the sample means of the variables ('EM'), or at a custom case. The marginal effects for dummy variables are actually the differences in expected outcomes for values of the dummy of 1 and 0. The marginal effects are displayed along with their statistical significance, evaluated based on the chosen 'at' parameter.

Value

If `verbose=TRUE` (default), prints the marginal effects, their standard errors, z-values, p-values, and significance levels.

If `verbose=FALSE`, returns a list containing the following components:

`where` A description of how the marginal effects have been evaluated.

`dEdq` The marginal effect. The partial derivative of the expected count with respect to a variable `q` in the `X` and or `Z` matrix, or the difference in expectation if `q` is binary.

`se` The standard errors of the marginal effects evaluated numerically and using a Jacobian via the delta method.

See Also

`dEdq_nb`, `dEdq_nb_noinfl`, `dEdq_pois`, `dEdq_pois_noinfl`, `model.frame`, `model.matrix`, `numericDeriv`

Examples

```
df <- data.frame(x = rnorm(100), z = rnorm(100), y = rpois(100, lambda = 5))
model <- oneinfl(y ~ x | z, df = df, dist = "Poisson")
margins(model, df, at = "AE") # Average Effect
margins(model, df, at = "EM", verbose=FALSE) # Effect at Means, suppress printing
margins(model, df, at = list(x = 1, z = 0)) # Custom case
```

`oneinfl` *One-Inflated Regression Model*

Description

Fits a one-inflated positive Poisson (OIPP) or one-inflated zero-truncated negative binomial (OIZTNB) regression model.

Usage

```
oneinfl(formula, df, dist = "negbin", start = NULL, method = "BFGS")
```

Arguments

`formula`	A symbolic description of the model to be fitted. Variables before the pipe '|' link to the usual Poisson rate parameter, after the pipe link to the one-inflation parameter.
`df`	A data frame containing the variables in the model.
`dist`	A character string specifying the distribution to use. Options are '"Poisson"' or '"negbin"'.
`start`	Optional. A numeric vector of starting values for the optimization process. Defaults to 'NULL', in which case starting values are attempted to be chosen automatically.

`method`	A character string specifying the optimization method to be passed to `optim`. Defaults to ‘"BFGS"‘.

Details

This function fits a regression model for one-inflated counts. One-inflated models are used when there are an excess number of ones, relative to a Poisson or negative binomial process.

The function supports two distributions: - ‘"Poisson"‘: One-inflated Poisson regression. - ‘"negbin"‘: One-inflated negative binomial regression.

The function uses numerical optimization via `optim` to estimate the parameters.

Value

An object of class ‘"oneinflmodel"‘ containing the following components:

`beta` Estimated coefficients for the rate component of the model.

`gamma` Estimated coefficients for the one-inflation component of the model.

`alpha` Dispersion parameter (only for negative binomial distribution).

`vc` Variance-covariance matrix of the estimated parameters.

`logl` Log-likelihood of the fitted model.

`avgw` Average one-inflation probability.

`absw` Mean absolute one-inflation probability.

`dist` The distribution used for the model ("Poisson" or "negbin").

`formula` The formula used for the model.

See Also

`summary` for summarizing the fitted model. `margins` for calculating the marginal effects of regressors. `oneWald` to test for no one-inflation. `signifWald` for testing the joint significance of a single regressor that appears before and after the pipe ‘|‘. `oneplot` for plotting actual and predicted counts. `predict` for expected response/dependent variable at each observation. `truncreg` for fitting positive Poisson (PP) and zero-truncated negative binomial (ZTNB) models. `oneLRT` to test for no one-inflation or no overdispersion using a nested PP, OIPP, or ZTNB model.

Examples

```
# Example usage
df <- data.frame(x = rnorm(100), z = rnorm(100), y = rpois(100, lambda = 1) + 1)
model <- oneinfl(y ~ x | z, df = df, dist = "Poisson")
summary(model)
margins(model, df)
oneWald(model)
predict(model, df=df)
```

oneLRT *Likelihood Ratio Test for Nested Models*

Description

Performs a likelihood ratio test (LRT) to compare two nested models estimated by `oneinfl` or `truncreg`. It calculates the LRT statistic and its associated p-value, testing whether the more complex model provides a significantly better fit to the data than the simpler model.

Usage

```
oneLRT(mod0, mod1)
```

Arguments

`mod0`	A model object (typically the simpler model) estimated using `oneinfl` or `truncreg`.
`mod1`	A model object (typically the more complex model) estimated using `oneinfl` or `truncreg`.

Details

The function extracts the log-likelihoods and number of parameters from the two models. It then calculates the LRT statistic:

$$-2 \cdot (\ell_0 - \ell_1)$$

where ℓ_0 and ℓ_1 are the log-likelihoods of the simpler and more complex models, respectively. The degrees of freedom for the test are equal to the difference in the number of parameters between the models.

The likelihood ratio test is commonly used to test for: - *Overdispersion*: Comparing a Poisson model to a negative binomial model. - *One-inflation*: Comparing a one-inflated model to a non-one-inflated model.

Value

A list with the following components:

`LRTstat` The likelihood ratio test statistic.

`pval` The p-value associated with the test statistic, based on a chi-squared distribution.

See Also

`oneinfl` for fitting one-inflated models. `truncreg` for fitting zero-truncated models. `pchisq` for the chi-squared distribution.

Examples

```
# Example: One-inflation test
df <- data.frame(y = rpois(100, lambda = 5), x = rnorm(100), z = rnorm(100))
OIZTNB <- oneinfl(y ~ x | z, df = df, dist = "negbin")
ZTNB <- truncreg(y ~ x, df = df, dist = "negbin")
oneLRT(OIZTNB, ZTNB)

# Example: Overdispersion test
OIPP <- oneinfl(y ~ x | z, df = df, dist = "Poisson")
oneLRT(OIZTNB, OIPP)
```

---

oneplot *Plot Observed Data and Model Predictions*

Description

Generates a bar plot of observed count data and overlays predicted values from one or more models fitted using `oneinfl` or `truncreg`.

Usage

```
oneplot(
  model1,
  model2 = NULL,
  model3 = NULL,
  model4 = NULL,
  df,
  maxpred = NULL,
  ylimit = NULL,
  ccex = 1.5
)
```

Arguments

`model1`	The first fitted model object, either a one-inflated model (class ‘"oneinflmodel"‘) or a truncated model (class ‘"truncmodel"‘).
`model2`	Optional. A second fitted model object, structured similarly to `model1`.
`model3`	Optional. A third fitted model object.
`model4`	Optional. A fourth fitted model object.
`df`	A data frame containing the variables used in the models.
`maxpred`	Optional. The maximum count value to include in the plot. Defaults to the maximum observed count.
`ylimit`	Optional. The upper limit for the y-axis. Defaults to 1.1 times the highest observed frequency.
`ccex`	Optional. A numeric value controlling the size of plot points and lines. Defaults to `1.5`.

Details

This function visualizes observed count data as a bar plot and overlays predicted values from up to four models. The function automatically detects the type of model (Poisson or negative binomial; one-inflated or truncated) and adjusts the plot accordingly. Predictions are generated using the `pred` function.

Model types are distinguished by different point and line styles:

- Poisson (PP): Dark magenta, triangle-down
- Zero-truncated negative binomial (ZTNB): Red, diamond
- One-inflated Poisson (OIPP): Green, triangle-up
- One-inflated zero-truncated negative binomial (OIZTNB): Blue, circle

The legend in the top-right corner of the plot indicates the models displayed.

Value

A plot is generated but no values are returned.

See Also

`oneinfl` for fitting one-inflated models. `truncreg` for fitting truncated models. `pred` for generating predictions used in the plot.

Examples

```
# Example usage
df <- data.frame(x = rnorm(100), z = rnorm(100), y = rpois(100, lambda = 5) + 1)
model1 <- oneinfl(y ~ x | z, df = df, dist = "Poisson")
model2 <- truncreg(y ~ x, df = df, dist = "Poisson")
oneplot(model1, model2, df = df, maxpred = 10)
```

`oneWald` *Wald Test for One-Inflation*

Description

Performs a Wald test to evaluate the significance of the one-inflation parameters in a model estimated using `oneinfl`.

Usage

```
oneWald(model)
```

Arguments

`model`	A model object of class `"oneinflmodel"` estimated using `oneinfl`. The model must include one-inflation parameters (`gamma`) and a variance-covariance matrix (`vc`).

Details

The Wald test evaluates the null hypothesis that all one-inflation parameters (`gamma`) are equal to zero, indicating no one-inflation. The test statistic is calculated as:

$$W = \gamma^\top V^{-1} \gamma$$

where γ is the vector of one-inflation parameters and V is their variance-covariance matrix. The p-value is computed using a chi-squared distribution with degrees of freedom equal to the length of γ.

This test is commonly used to determine whether a one-inflated model provides a significantly better fit than a non-one-inflated counterpart.

Value

A list with the following components:

`W` The Wald test statistic.

`pval` The p-value associated with the test statistic, based on a chi-squared distribution.

See Also

`oneinfl` for fitting one-inflated models. `oneLRT` for a likelihood ratio test of nested models. `pchisq` for the chi-squared distribution.

Examples

```
# Example usage
df <- data.frame(y = rpois(100, lambda = 5), x = rnorm(100), z = rnorm(100))
OIZTNB <- oneinfl(y ~ x | z, df = df, dist = "negbin")
oneWald(OIZTNB)
```

`predict.oneinflmodel` *Predicted Expected Response for One-Inflated or Truncated Models*

Description

Calculates the predicted expected response for a model fitted using `oneinfl` or `truncreg`.

Usage

```
## S3 method for class 'oneinflmodel'
predict(object, ...)
```

Arguments

`object`	An object of class ‘oneinflmodel‘
`...`	Additional argument ‘df‘, a data frame used to calculate the expected value of the response variable.

Details

This function computes the expected response based on the fitted model. The computation differs depending on the distribution. For `Poisson (OIPP)`, predicted values are computed using `E_pois`. For `Negative Binomial (OIZTNB)`, predicted values are computed using `E_negbin`.

Value

A numeric vector of predicted expected responses for the observations in `df`.

See Also

`oneinfl` for fitting one-inflated models. `E_pois`, `E_negbin`, for the expected value calculations.

Examples

```
# Example usage
df <- data.frame(x = rnorm(100), z = rnorm(100), y = rpois(100, lambda = 5))
model <- oneinfl(y ~ x | z, df = df, dist = "Poisson")
predict(model, df = df)
```

`predict.truncmodel` *Predicted Expected Response for One-Inflated or Truncated Models*

Description

Calculates the predicted expected response for a model fitted using `oneinfl` or `truncreg`.

Usage

```
## S3 method for class 'truncmodel'
predict(object, ...)
```

Arguments

`object`	An object of class ‘truncmodel‘
`...`	Additional argument ‘df‘, a data frame used to calculate the expected value of the response variable.

Details

This function computes the expected response based on the fitted model. The computation differs depending on the distribution. For `Poisson (PP)`, predicted values are computed using `E_pois_noinfl`. For `Negative Binomial (ZTNB)`, predicted values are computed using `E_negbin_noinfl`.

Value

A numeric vector of predicted expected responses for the observations in `df`.

See Also

`oneinfl` for fitting one-inflated models. `truncreg` for fitting truncated models. `E_pois_noinfl`, `E_negbin_noinfl` for the expected value calculations.

Examples

```
# Example usage
df <- data.frame(x = rnorm(100), y = rpois(100, lambda = 5))
model <- truncreg(y ~ x, df = df, dist = "Poisson")
predict(model, df = df)
```

`roipp` *Generate Random Counts from a One-Inflated Poisson Process*

Description

Simulates count data from a one-inflated Poisson process using specified parameters for the rate and one-inflation components.

Usage

```
roipp(b, g, X, Z)
```

Arguments

`b`	A numeric vector of coefficients for the rate component.
`g`	A numeric vector of coefficients for the one-inflation component.
`X`	A matrix or data frame of predictor variables for the rate component.
`Z`	A matrix or data frame of predictor variables for the one-inflation component.

Details

This function generates count data from a one-inflated Poisson process. The process combines:

- A Poisson distribution for counts greater than one.
- A one-inflation component that adjusts the probability of observing a count of one.

The algorithm:

1. Calculates the rate parameter (λ) as $\exp(X \cdot b)$.
2. Computes the one-inflation probabilities (ω) based on $Z \cdot g$.
3. Simulates counts for each observation:
   - Draws a random number to determine whether the count is one.
   - Iteratively calculates probabilities for higher counts until the random number is matched.

This function is useful for generating synthetic data for testing or simulation studies involving one-inflated Poisson models.

Value

A numeric vector of simulated count data.

See Also

`oneinfl` for fitting one-inflated models.

Examples

```
# Example usage
set.seed(123)
X <- matrix(rnorm(100), ncol = 2)
Z <- matrix(rnorm(100), ncol = 2)
b <- c(0.5, -0.2)
g <- c(1.0, 0.3)
simulated_data <- roipp(b, g, X, Z)
print(simulated_data)
```

`roiztnb` *Generate Random Counts from a One-Inflated Zero-Truncated Negative Binomial Process*

Description

Simulates count data from a one-inflated, zero-truncated negative binomial (OIZTNB) process using specified parameters for the rate, one-inflation, and dispersion components.

Usage

```
roiztnb(b, g, alpha, X, Z)
```

Arguments

`b`	A numeric vector of coefficients for the rate component.
`g`	A numeric vector of coefficients for the one-inflation component.
`alpha`	A numeric value representing the dispersion parameter for the negative binomial distribution.
`X`	A matrix or data frame of predictor variables for the rate component.
`Z`	A matrix or data frame of predictor variables for the one-inflation component.

Details

This function generates count data from a one-inflated, zero-truncated negative binomial process. The process combines:

- A negative binomial distribution for counts greater than one.
- A one-inflation component that adjusts the probability of observing a count of one.

The algorithm:

1. Calculates the rate parameter (λ) as $\exp(X \cdot b)$.
2. Computes the one-inflation probabilities (ω) based on $Z \cdot g$.
3. Computes the negative binomial dispersion parameter ($\theta = \lambda/\alpha$).
4. Simulates counts for each observation:
   - Draws a random number to determine whether the count is one.
   - Iteratively calculates probabilities for higher counts until the random number is matched.

This function is useful for generating synthetic data for testing or simulation studies involving one-inflated, zero-truncated negative binomial models.

Value

A numeric vector of simulated count data.

See Also

`oneinfl` for fitting one-inflated models.

Examples

```
# Example usage
set.seed(123)
X <- matrix(rnorm(100), ncol = 2)
Z <- matrix(rnorm(100), ncol = 2)
b <- c(0.5, -0.2)
g <- c(1.0, 0.3)
alpha <- 1.5
simulated_data <- roiztnb(b, g, alpha, X, Z)
print(simulated_data)
```

`rpp` *Generate Random Counts from a Zero-Truncated Poisson Process*

Description

Simulates count data from a zero-truncated Poisson process using specified parameters for the rate component.

Usage

```
rpp(b, X)
```

Arguments

`b`	A numeric vector of coefficients for the rate component.
`X`	A matrix or data frame of predictor variables for the rate component.

Details

This function generates count data from a zero-truncated Poisson process, which models count data without zeros. The process involves:

- Calculating the rate parameter (λ) as $\exp(X \cdot b)$.
- Iteratively computing probabilities for counts starting from 1 and adding to the cumulative probability until a randomly drawn value is matched.

This function is useful for generating synthetic data for testing or simulation studies involving zero-truncated Poisson models.

Value

A numeric vector of simulated count data.

See Also

`truncreg` for fitting zero-truncated models.

Examples

```
# Example usage
set.seed(123)
X <- matrix(rnorm(100), ncol = 2)
b <- c(0.5, -0.2)
simulated_data <- rpp(b, X)
print(simulated_data)
```

signifWald *Wald Test for Significance of a Predictor Variable*

Description

Performs a Wald test to evaluate the joint significance of a predictor variable in both the rate and one-inflation components of a model.

Usage

```
signifWald(model, varname)
```

Arguments

`model`	A fitted model object of class `"oneinflmodel"`.
`varname`	A character string specifying the name of the predictor variable to test.

Details

This function tests the null hypothesis that the coefficients for the specified predictor variable are jointly equal to zero in both the rate (`beta`) and one-inflation (`gamma`) components of the model. The test statistic is calculated as:

$$W = \mathbf{c}^\top V^{-1}\mathbf{c}$$

where **c** is the vector of coefficients for the predictor in the rate and one-inflation components, and V is their variance-covariance matrix. The p-value is computed using a chi-squared distribution with 2 degrees of freedom.

Value

A list with the following components:

`W` The Wald test statistic.

`pval` The p-value associated with the test statistic, based on a chi-squared distribution with 2 degrees of freedom.

See Also

`oneinfl` for fitting one-inflated models. `oneWald` for a general Wald test of one-inflation parameters.

Examples

```
# Example usage
set.seed(123)
df <- data.frame(x = rnorm(100), z = rnorm(100), y = rpois(100, lambda = 5))
model <- oneinfl(y ~ x | z, df = df, dist = "Poisson")
result <- signifWald(model, varname = "x")
```

```
print(result$W)    # Wald test statistic
print(result$pval) # p-value
```

summary.oneinflmodel *Summarize One-Inflated Regression Models*

Description

Provides a summary of the fitted model, including estimated coefficients, standard errors, significance levels, and other relevant statistics.

Usage

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

Arguments

`object`	A model object of class ‘"oneinflmodel"‘ (for one-inflated models).
`...`	Additional arguments (currently unused).

Details

This function generates a detailed summary of the fitted model, including: - Estimated coefficients for the rate component (`beta`). - Estimated coefficients for the one-inflation component (`gamma`). - Standard errors. - z-statistics, associated p-values, and corresponding significance codes. - Average, and average absolute, one-inflation. - Log-likelihood of the fitted model.

Value

Prints a summary table of coefficients, standard errors, z-values, p-values, significance codes, one-inflation probabilities, and log-likelihood.

See Also

`oneinfl` for fitting one-inflated models.

Examples

```
# Example usage
df <- data.frame(x = rnorm(100), z = rnorm(100), y = rpois(100, lambda = 5))
model <- oneinfl(y ~ x | z, df = df, dist = "Poisson")
summary(model)
```

`summary.truncmodel` *Summarize Truncated Regression Models*

Description

Provides a summary of the fitted model, including estimated coefficients, standard errors, significance levels, and other relevant statistics.

Usage

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

Arguments

`object`	A model object of class ‘"truncmodel"‘ (for truncated models).
`...`	Additional arguments (currently unused).

Details

This function generates a detailed summary of the fitted model, including: - Estimated coefficients for the rate component (`beta`). - Standard errors. - z-statistics, associated p-values, and corresponding significance codes. - Log-likelihood of the fitted model.

Value

Prints a summary table of coefficients, standard errors, z-values, p-values, significance codes, and log-likelihood.

See Also

`truncreg` for fitting truncated regression models.

Examples

```
# Example usage
df <- data.frame(x = rnorm(100), y = rpois(100, lambda = 5))
model <- truncreg(y ~ x, df = df, dist = "Poisson")
summary(model)
```

`truncreg` *Truncated Regression Model*

Description

Fits a positive Poisson (PP) or zero-truncated negative binomial (ZTNB) regression model.

Usage

```
truncreg(formula, df, dist = "negbin", start = NULL, method = "BFGS")
```

Arguments

`formula`	A symbolic description of the model to be fitted.
`df`	A data frame containing the variables in the model.
`dist`	A character string specifying the distribution to use. Options are ‘"Poisson"‘ or ‘"negbin"‘.
`start`	Optional. A numeric vector of starting values for the optimization process. Defaults to ‘NULL‘, in which case starting values are attempted to be chosen automatically.
`method`	A character string specifying the optimization method to be passed to `optim`. Defaults to ‘"BFGS"‘.

Details

This function fits a regression model for zero-truncated counts. Zero-truncated models are used when the count data does not include zeros, such as in cases where only positive counts are observed.

The function supports two distributions: - ‘"Poisson"‘: Zero-truncated Poisson regression. - ‘"negbin"‘: Zero-truncated negative binomial regression.

The function uses numerical optimization via `optim` to estimate the parameters.

Value

An object of class ‘"truncmodel"‘ containing the following components:

`beta` Estimated coefficients for the regression model.

`alpha` Dispersion parameter (only for negative binomial distribution).

`vc` Variance-covariance matrix of the estimated parameters.

`logl` Log-likelihood of the fitted model.

`dist` The distribution used for the model ("Poisson" or "negbin").

`formula` The formula used for the model.

See Also

`summary` for summarizing the fitted model.

Examples

```
# Example usage
df <- data.frame(x = rnorm(100), y = rpois(100, lambda = 1) + 1)
model <- truncreg(y ~ x, df = df, dist = "Poisson")
summary(model)
```

Index