Package 'COMPoissonReg'

May 3, 2017

Type Package

Title Conway-Maxwell Poisson (COM-Poisson) Regression

Version 0.4.1

Date 2017-05-03

Author Kimberly Sellers <kfs7@georgetown.edu>
Thomas Lotze <thomas.lotze@thomaslotze.com>
Andrew Raim <andrew.raim@gmail.com>

Maintainer Andrew Raim <andrew.raim@gmail.com>

URL https://github.com/lotze/COMPoissonReg

Description Fit Conway-Maxwell Poisson (COM-Poisson or CMP) regression models to count data (Sellers & Shmueli, 2010) <doi:10.1214/09-AOAS306>. The package provides functions for model estimation, dispersion testing, and diagnostics. Zero-inflated CMP regression (Sellers & Raim, 2016) <doi:10.1016/j.csda.2016.01.007> is also supported.

License GPL-2 | GPL-3

LazyLoad yes

Depends stats

NeedsCompilation no

Repository CRAN

Date/Publication 2017-05-03 12:03:19 UTC

R topics documented:

`COMPoissonReg-package` *Estimate parameters for COM-Poisson regression*

Description

This package offers the ability to compute the parameter estimates for a COM-Poisson or zero-inflated (ZI) COM-Poisson regression and associated standard errors. This package also provides a hypothesis test for determining statistically significant data dispersion, and other model diagnostics.

Details

This package offers the ability to compute the COM-Poisson parameter estimates and associated standard errors for a regular regression model or a zero-inflated regression model (via the glm.cmp function).

Further, the user can perform a hypothesis test to determine the statistically significant need for using COM-Poisson regression to model the data. The test addresses the matter of statistically significant dispersion.

The main order of functions for COM-Poisson regression is as follows:

1. Compute Poisson estimates (using glm for Poisson regression or pscl for ZIP regression)
2. Use Poisson estimates as starting values to determine COM-Poisson estimates (using glm.cmp)
3. Compute associated standard errors (using sdev function)

From here, there are lots of ways to proceed, so order is irrelevant:

- Perform a hypothesis test to assess for statistically significant dispersion (using equitest or parametric_bootstrap)
- Compute leverage (using leverage) and deviance (using deviance)
- Predict the outcome for new examples, using predict

The package also supports fitting of the zero-inflated COM-Poisson model (ZICMP). Most of the tools available for COM-Poisson are also available for ZICMP.

Author(s)

Kimberly Sellers, Thomas Lotze (Maintainer, <thomas.lotze@thomaslotze.com>), Andrew M. Raim

References

Kimberly F. Sellers & Galit Shmueli (2010). A Flexible Regression Model for Count Data. Annals of Applied Statistics, 4(2), 943-961.

Kimberly F. Sellers and Andrew M. Raim (2016). A Flexible Zero-Inflated Model to Address Data Dispersion. Computational Statistics and Data Analysis, 99, 68-80.

Examples

```
## load freight data
data(freight)

# Fit standard Poisson model
glm.out <- glm(broken ~ transfers, data=freight,
  family=poisson, na.action=na.exclude)
print(glm.out)

# Fit COM-Poisson model (with intercept-only regression linked to the
# dispersion parameter)
cmp.out <- glm.cmp(broken ~ transfers, data=freight)
print(cmp.out)
coef(cmp.out)
nu(cmp.out)[1]

# Compute associated standard errors
sdev(cmp.out)

# Likelihood ratio test for dispersion parameter
# Test for H_0: dispersion equal to 1 vs. H_1: not equal to 1
# (i.e. Poisson vs. COM-Poisson regression models)
lrt <- equitest(cmp.out)

# Compute constant COM-Poisson leverage
lev <- leverage(cmp.out)


# Compute constant COM-Poisson deviances
dev <- deviance(cmp.out)


# Compute fitted values
y.hat <- predict(cmp.out, newdata=freight)

# Compute residual values
res <- residuals(cmp.out)
print(summary(res))

# Compute MSE
mean(res^2)

# Compute predictions on new data
new_data <- data.frame(transfers=(0:10))
y.hat <- predict(cmp.out, newdata=new_data)
plot(0:10, y.hat, type="l",
  xlab="number of transfers", ylab="predicted number broken")


# Compute parametric bootstrap results and use them to generate
# 0.95 confidence intervals for parameters.
cmp.boot <- parametric_bootstrap(cmp.out, reps=1000)
```

```
print(apply(cmp.boot, 2, quantile, c(0.025,0.975)))



## load couple data
data(couple)

# Fit standard Poisson model
glm.out <- glm(UPB ~ EDUCATION + ANXIETY, data=couple, family=poisson)
print(glm.out)

# Fit ZICMP model
zicmp.out <- glm.cmp(UPB ~ EDUCATION + ANXIETY,
  formula.nu = ~ 1,
  formula.p = ~ EDUCATION + ANXIETY,
  data=couple)
print(zicmp.out)

# Compute standard errors for estimates of coefficients
sdev(zicmp.out)

# Likelihood ratio test for equidispersion (H0: nu = 1 vs H1: not)
equitest(zicmp.out)

# Compute fitted values
y.hat <- predict(zicmp.out)

# Compute residuals
res.raw <- residuals(zicmp.out, type = "raw")
res.quan <- residuals(zicmp.out, type = "quantile")
print(summary(res.raw))
print(summary(res.quan))

# Compute predictions on new data
new_data <- data.frame(EDUCATION = round(1:20 / 20), ANXIETY = seq(-3,3, length.out = 20))
y.hat.new <- predict(zicmp.out, newdata=new_data)
print(y.hat.new)

# Compute parametric bootstrap results and use them to generate
# 0.95 confidence intervals for parameters.
zicmp.boot <- parametric_bootstrap(zicmp.out, reps=1000)
print(apply(zicmp.boot, 2, quantile, c(0.025,0.975)))
```

CMP Distribution *COM-Poisson Distribution*

Description

Functions for the COM-Poisson distribution.

Usage

```
dcmp(x, lambda, nu, z = NULL, log = FALSE, max = 100)
pcmp(x, lambda, nu, max = 100)
qcmp(q, lambda, nu, max = 100, log.p = FALSE)
rcmp(n, lambda, nu, max = 100)
```

Arguments

x	vector of quantiles.
q	vector of probabilities.
n	number of observations.
z	normalizing constant. Can be passed in to save computation; otherwise computed internally.
lambda	rate parameter.
nu	dispersion parameter.
max	maximum number to use for truncating infinite sums.
log, log.p	logical; if TRUE, probabilities p are given as log(p).

Value

dcmp gives the density, pcmp gives the cumulative probability, qcmp gives the quantile function, and rcmp generates random values.

Author(s)

Kimberly Sellers

References

Kimberly F. Sellers & Galit Shmueli (2010). A Flexible Regression Model for Count Data. Annals of Applied Statistics, 4(2), 943-961.

---

couple.rda *Couple dataset*

Description

A dataset investigating the impact of education level and level of anxious attachment on unwanted pursuit behaviors in the context of couple separation.

Usage

```
data(couple)
```

Format

- UPB = number of unwanted pursuit behavior perpetrations.
- EDUCATION = 1 if at least bachelor's degree; 0 otherwise.
- ANXIETY = continuous measure of anxious attachment.

References

Loeys, T., Moerkerke, B., DeSmet, O., Buysse, A., 2012. The analysis of zero-inflated count data: Beyond zero-inflated Poisson regression. British J. Math. Statist. Psych. 65 (1), 163-180.

---

equitest *Likelihood ratio test for Equidispersion*

Description

A generic function for the likelihood ratio test for equidispersion using the output of a fitted mode. The function invokes particular methods which depend on the class of the first argument.

Usage

```
equitest(object, ...)
```

Arguments

`object`	a model object
`...`	other parameters which might be required by the model

Details

See the documentation of the particular methods for details.

Value

Returns the test statistic and p-value determined from the χ^2_1 distribution.

Author(s)

Thomas Lotze

See Also

`equitest.cmp`, `equitest.zicmp`

freight.rda *Freight dataset*

Description

A set of data on airfreight breakage (breakage of ampules filled with some biological substance are shipped in cartons).

Usage

```
data(freight)
```

Format

- broken = number of ampules found broken upon arrival.
- transfers = number of times carton was transferred from one aircraft to another.

References

Kutner MH, Nachtsheim CJ, Neter J (2003). Applied Linear Regression Models, Fourth Edition. McGraw-Hill.

glm.cmp *COM-Poisson and Zero-Inflated COM-Poisson regression*

Description

Fit COM-Poisson regression using maximum likelihood estimation. Zero-Inflated COM-Poisson can be fit by specifying a regression for the overdispersion parameter.

The COM-Poisson regression model is

$$y_i \sim \mathrm{CMP}(\lambda_\mathrm{i}, \nu_\mathrm{i}), \quad \log \lambda_\mathrm{i} = \mathrm{x_i}^\top \beta, \quad \log \nu_\mathrm{i} = \mathrm{s_i}^\top \gamma.$$

The Zero-Inflated COM-Poisson regression model assumes that y_i is 0 with probability p_i or y_i^* with probability $1 - p_i$, where

$$y_i^* \sim \mathrm{CMP}(\lambda_\mathrm{i}, \nu_\mathrm{i}), \quad \log \lambda_\mathrm{i} = \mathrm{x_i}^\top \beta, \quad \log \nu_\mathrm{i} = \mathrm{s_i}^\top \gamma, \quad \log \mathrm{p_i} = \mathrm{w_i}^\top \zeta.$$

Usage

```
glm.cmp(formula.lambda, formula.nu = ~ 1, formula.p = NULL,
        beta.init = NULL, gamma.init = NULL, zeta.init = NULL, max = 100, ...)

## S3 method for class 'cmp'
AIC(object, ..., k = 2)
## S3 method for class 'cmp'
BIC(object, ...)
## S3 method for class 'cmp'
coef(object, ...)
## S3 method for class 'cmp'
deviance(object, ...)
## S3 method for class 'cmp'
equitest(object, ...)
## S3 method for class 'cmp'
leverage(object, ...)
## S3 method for class 'cmp'
logLik(object, ...)
## S3 method for class 'cmp'
nu(object, ...)
## S3 method for class 'cmp'
parametric_bootstrap(object, reps = 1000, report.period = reps + 1, ...)
## S3 method for class 'cmp'
predict(object, newdata = NULL, ...)
## S3 method for class 'cmp'
print(x, ...)
## S3 method for class 'cmp'
residuals(object, type = c("raw", "quantile"), ...)
## S3 method for class 'cmp'
sdev(object, ...)
## S3 method for class 'cmp'
summary(object, ...)


## S3 method for class 'zicmp'
AIC(object, ..., k = 2)
## S3 method for class 'zicmp'
BIC(object, ...)
## S3 method for class 'zicmp'
coef(object, ...)
## S3 method for class 'zicmp'
deviance(object, ...)
## S3 method for class 'zicmp'
equitest(object, ...)
## S3 method for class 'zicmp'
leverage(object, ...)
## S3 method for class 'zicmp'
logLik(object, ...)
```

```
## S3 method for class 'zicmp'
nu(object, ...)
## S3 method for class 'zicmp'
parametric_bootstrap(object, reps = 1000, report.period = reps + 1, ...)
## S3 method for class 'zicmp'
predict(object, newdata = NULL, ...)
## S3 method for class 'zicmp'
print(x, ...)
## S3 method for class 'zicmp'
residuals(object, type = c("raw", "quantile"), ...)
## S3 method for class 'zicmp'
sdev(object, ...)
## S3 method for class 'zicmp'
summary(object, ...)
```

Arguments

formula.lambda	regression formula linked to log(lambda)
formula.nu	regression formula linked to log(nu). By default, is taken to be intercept only.
formula.p	regression formula linked to logit(p). If NULL (the default), zero-inflation term is excluded from the model.
beta.init	initial values for regression coefficients of lambda.
gamma.init	initial values for regression coefficients of nu.
zeta.init	initial values for regression coefficients of p.
max	maximum number to use for truncating infinite sums.
...	other model parameters, such as data
object	object of type 'cmp' or 'zicmp'.
x	object of type 'cmp' or 'zicmp'.
k	Penalty per parameter to be used in AIC calculation.
newdata	New covariates to be used for prediction.
type	Type of residual to be computed.
reps	Number of bootstrap repetitions.
report.period	Report progress every report.period iterations.

Value

glm.cmp produces an object of either class 'cmp' or 'zicmp', depending on whether zero-inflation is used in the model. From this object, coefficients and other information can be extracted.

Author(s)

Kimberly Sellers, Thomas Lotze, Andrew Raim

References

Kimberly F. Sellers & Galit Shmueli (2010). A Flexible Regression Model for Count Data. Annals of Applied Statistics, 4(2), 943-961.

Kimberly F. Sellers and Andrew M. Raim (2016). A Flexible Zero-Inflated Model to Address Data Dispersion. Computational Statistics and Data Analysis, 99, 68-80.

`leverage` *Leverage*

Description

a generic function for the leverage of points used in various model fitting functions. The function invokes particular methods which depend on the class of the first argument.

Usage

```
leverage(object, ...)
```

Arguments

`object`	a model object
`...`	other parameters which might be required by the model

Details

See the documentation of the particular methods for details.

Value

The form of the value returned depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

Author(s)

Thomas Lotze

See Also

`leverage.cmp`

nu *Estimate for dispersion parameter*

Description

a generic function for the dispersion parameter estimate from the results of various model fitting functions. The function invokes particular methods which depend on the class of the first argument.

Usage

```
nu(object, ...)
```

Arguments

object	a model object
...	other parameters which might be required by the model

Details

See the documentation of the particular methods for details.

Value

The form of the value returned depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

Author(s)

Thomas Lotze

See Also

nu.cmp

parametric_bootstrap *Parametric Bootstrap*

Description

a generic function for the parametric bootstrap from the results of various model fitting functions. The function invokes particular methods which depend on the class of the first argument.

Usage

```
parametric_bootstrap(object, reps = 1000, report.period = reps+1, ...)
```

Arguments

`object`	a model object
`...`	other parameters which might be required by the model
`reps`	Number of bootstrap repetitions.
`report.period`	Report progress every `report.period` iterations.

Details

See the documentation of the particular methods for details.

Value

The form of the value returned depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

Author(s)

Thomas Lotze

See Also

`parametric_bootstrap.cmp`, `parametric_bootstrap.zicmp`

---

sdev *Standard deviations*

Description

a generic function for the standard deviation estimates from the results of various model fitting functions. The function invokes particular methods which depend on the class of the first argument.

Usage

```
sdev(object, ...)
```

Arguments

`object`	a model object
`...`	other parameters which might be required by the model

Details

See the documentation of the particular methods for details.

Value

The form of the value returned depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

Author(s)

Thomas Lotze

See Also

`sdev.cmp`, `sdev.zicmp`

`ZICMP Distribution` *ZICMP Distribution*

Description

Computes the density, cumulative probability, quantiles, and random draws for the zero-inflated COM-Poisson distribution.

Usage

```
dzicmp(x, lambda, nu, p, z = NULL, max = 100, log = FALSE)
pzicmp(x, lambda, nu, p, max = 100)
qzicmp(q, lambda, nu, p, max = 100, log.p = FALSE)
rzicmp(n, lambda, nu, p, max = 100)
```

Arguments

`x`	vector of quantiles.
`q`	vector of probabilities.
`n`	number of observations.
`z`	normalizing constant. Can be passed in to save computation; otherwise computed internally.
`lambda`	rate parameter.
`nu`	dispersion parameter.
`p`	zero-inflation probability parameter.
`max`	maximum number to use for truncating infinite sums.
`log, log.p`	logical; if TRUE, probabilities p are given as log(p).

Value

`dzicmp` gives the density, `pzicmp` gives the cumulative probability, `qzicmp` gives the quantile value, and `rzicmp` generates random numbers.

Author(s)

Kimberly Sellers, Andrew Raim

References

Kimberly F. Sellers and Andrew M. Raim (2016). A Flexible Zero-Inflated Model to Address Data Dispersion. Computational Statistics and Data Analysis, 99, 68-80.

Index