ZERO-INFLATED NEGATIVE BINOMIAL REGRESSION | R DATA ANALYSIS EXAMPLES

FROM: <https://stats.idre.ucla.edu/r/dae/zinb/>

Zero-inflated negative binomial regression is for modeling count variables with excessive zeros and it is usually for overdispersed count outcome variables. Furthermore, theory suggests that the excess zeros are generated by a separate process from the count values and that the excess zeros can be modeled independently.

This page uses the following packages. Make sure that you can load them before trying to run the examples on this page. If you do not have a package installed, run: install.packages("packagename"), or if you see the version is out of date, run: update.packages().

**require**(ggplot2)

**require**(pscl)

**require**(MASS)

**require**(boot)

**Version info:**Code for this page was tested in R version 3.1.1 (2014-07-10) On: 2014-08-11 With: boot 1.3-11; knitr 1.6; pscl 1.04.4; vcd 1.3-1; gam 1.09.1; coda 0.16-1; mvtnorm 1.0-0; GGally 0.4.7; plyr 1.8.1; MASS 7.3-33; Hmisc 3.14-4; Formula 1.1-2; survival 2.37-7; psych 1.4.5; reshape2 1.4; msm 1.4; phia 0.1-5; RColorBrewer 1.0-5; effects 3.0-0; colorspace 1.2-4; lattice 0.20-29; pequod 0.0-3; car 2.0-20; ggplot2 1.0.0

**Please note:** The purpose of this page is to show how to use various data analysis commands. It does not cover all aspects of the research process which researchers are expected to do. In particular, it does not cover data cleaning and checking, verification of assumptions, model diagnostics or potential follow-up analyses.

Examples of zero-inflated negative binomial regression

Example 1. School administrators study the attendance behavior of high school juniors at two schools. Predictors of the number of days of absence include gender of the student and standardized test scores in math and language arts.

Example 2. The state wildlife biologists want to model how many fish are being caught by fishermen at a state park. Visitors are asked how long they stayed, how many people were in the group, were there children in the group and how many fish were caught. Some visitors do not fish, but there is no data on whether a person fished or not. Some visitors who did fish did not catch any fish so there are excess zeros in the data because of the people that did not fish.

Description of the Data

Let’s pursue Example 2 from above.

We have data on 250 groups that went to a park. Each group was questioned about how many fish they caught (count), how many children were in the group (child), how many people were in the group (persons), and whether or not they brought a camper to the park (camper).

In addition to predicting the number of fish caught, there is interest in predicting the existence of excess zeros, i.e., the probability that a group caught zero fish. We will use the variables child, persons, andcamper in our model. Let’s look at the data.

zinb <- **read.csv**("https://stats.idre.ucla.edu/stat/data/fish.csv")

zinb <- **within**(zinb, {

nofish <- **factor**(nofish)

livebait <- **factor**(livebait)

camper <- **factor**(camper)

})

**summary**(zinb)

## nofish livebait camper persons child xb

## 0:176 0: 34 0:103 Min. :1.00 Min. :0.000 Min. :-3.275

## 1: 74 1:216 1:147 1st Qu.:2.00 1st Qu.:0.000 1st Qu.: 0.008

## Median :2.00 Median :0.000 Median : 0.955

## Mean :2.53 Mean :0.684 Mean : 0.974

## 3rd Qu.:4.00 3rd Qu.:1.000 3rd Qu.: 1.964

## Max. :4.00 Max. :3.000 Max. : 5.353

## zg count

## Min. :-5.626 Min. : 0.0

## 1st Qu.:-1.253 1st Qu.: 0.0

## Median : 0.605 Median : 0.0

## Mean : 0.252 Mean : 3.3

## 3rd Qu.: 1.993 3rd Qu.: 2.0

## Max. : 4.263 Max. :149.0

*## histogram with x axis in log10 scale*

**ggplot**(zinb, **aes**(count, fill = camper)) +

**geom\_histogram**() +

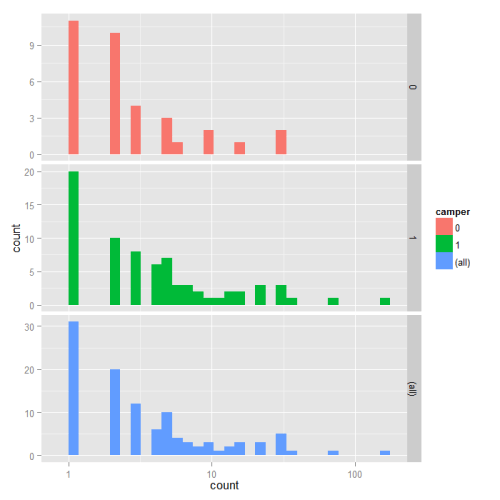
**scale\_x\_log10**() +

**facet\_grid**(camper ~ ., margins=TRUE, scales="free\_y")

## stat\_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.

## stat\_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.

## stat\_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.



Analysis methods you might consider

Before we show how you can analyze this with a zero-inflated negative binomial analysis, let’s consider some other methods that you might use.

* OLS Regression – You could try to analyze these data using OLS regression. However, count data are highly non-normal and are not well estimated by OLS regression.
* Zero-inflated Poisson Regression – Zero-inflated Poisson regression does better when the data is not overdispersed, i.e. when variance is not much larger than the mean.
* Ordinary Count Models – Poisson or negative binomial models might be more appropriate if there are not excess zeros.

Zero-inflated negative binomial regression

A zero-inflated model assumes that zero outcome is due to two different processes. For instance, in the example of fishing presented here, the two processes are that a subject has gone fishing vs. not gone fishing. If not gone fishing, the only outcome possible is zero. If gone fishing, it is then a count process. The two parts of the a zero-inflated model are a binary model, usually a logit model to model which of the two processes the zero outcome is associated with and a count model, in this case, a negative binomial model, to model the count process. The expected count is expressed as a combination of the two processes. Taking the example of fishing again:

E(nfish caught=k)=P(not gone fishing)∗0+P(gone fishing)∗E(y=k|gone fishing)E(nfish caught=k)=P(not gone fishing)∗0+P(gone fishing)∗E(y=k|gone fishing)

To understand the zero-inflated negative binomial regression, let’s start with the negative binomial model. There are multiple parameterizations of the negative binomial model, we focus on NB2. The negative binomial probability density function is:

PDF(y;p,r)=(yi+r–1)!yi!(r−1)!pri(1–pi)yiPDF(y;p,r)=(yi+r–1)!yi!(r−1)!pir(1–pi)yi

where (p) is the probability of (r) successes. From this we can derive the likelihood function, which is given by:

L(μ;y,α)=∏i=1nexp(yiln(αμi1+αμi)−1αln(1+αμi)+lnΓ(yi+1α)−lnΓ(yi+1)–lnΓ(1α))L(μ;y,α)=∏i=1nexp(yiln(αμi1+αμi)−1αln(1+αμi)+lnΓ(yi+1α)−lnΓ(yi+1)–lnΓ(1α))

here we find the likelihood of the expected value, μμ given the data and αα which allows for dispersion. Typically, this would be expressed as a log likelihood, denoted by script L, LL:

L(μ;y,α)=∑i=1nyiln(αμi1+αμi)−1αln(1+αμi)+lnΓ(yi+1α)−lnΓ(yi+1)–lnΓ(1α)L(μ;y,α)=∑i=1nyiln(αμi1+αμi)−1αln(1+αμi)+lnΓ(yi+1α)−lnΓ(yi+1)–lnΓ(1α)

which can be expressed in terms of our model by replacing μiμi with exp(x′iβ)exp(xi′β). Turning to the zero-inflated negative binomial model, the expression of the likelihood function depends on whether the observed value is a zero or greater than zero. From the logistic model of yi>1yi>1 versus y=0y=0:

p=11+e−x′iβp=11+e−xi′β

and

1–p=11+ex′iβ1–p=11+exi′β

then

L=⎧⎩⎨⎪⎪⎪⎪∑ni=1[ln(pi)+(1–pi)(11+αμi)1α]∑ni=1[ln(pi)+lnΓ(1α+yi)–lnΓ(yi+1)–lnΓ(1α)+(1α)ln(11+αμi)+yiln(1–11+αμi)]if yi=0if yi>0L={∑i=1n[ln(pi)+(1–pi)(11+αμi)1α]if yi=0∑i=1n[ln(pi)+lnΓ(1α+yi)–lnΓ(yi+1)–lnΓ(1α)+(1α)ln(11+αμi)+yiln(1–11+αμi)]if yi>0

Finally, note that R does not estimate αα but θθ, the inverse of αα.

Now let’s build up our model. We are going to use the variables child and camper to model the count in the part of negative binomial model and the variable persons in the logit part of the model. We use the pscl to run a zero-inflated negative binomial regression. We begin by estimating the model with the variables of interest.

m1 <- **zeroinfl**(count ~ child + camper | persons,

data = zinb, dist = "negbin", EM = TRUE)

**summary**(m1)

##

## Call:

## zeroinfl(formula = count ~ child + camper | persons, data = zinb,

## dist = "negbin", EM = TRUE)

##

## Pearson residuals:

## Min 1Q Median 3Q Max

## -0.586 -0.462 -0.389 -0.197 18.013

##

## Count model coefficients (negbin with log link):

## Estimate Std. Error z value Pr(>|z|)

## (Intercept) 1.371 0.256 5.35 8.6e-08 \*\*\*

## child -1.515 0.196 -7.75 9.4e-15 \*\*\*

## camper1 0.879 0.269 3.26 0.0011 \*\*

## Log(theta) -0.985 0.176 -5.60 2.1e-08 \*\*\*

##

## Zero-inflation model coefficients (binomial with logit link):

## Estimate Std. Error z value Pr(>|z|)

## (Intercept) 1.603 0.836 1.92 0.055 .

## persons -1.666 0.679 -2.45 0.014 \*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Theta = 0.373

## Number of iterations in BFGS optimization: 2

## Log-likelihood: -433 on 6 Df

The output looks very much like the output from two OLS regressions in R.

Below the model call, you will find a block of output containing negative binomial regression coefficients for each of the variables along with standard errors, z-scores, and p-values for the coefficients. A second block follows that corresponds to the inflation model. This includes logit coefficients for predicting excess zeros along with their standard errors, z-scores, and p-values.

All of the predictors in both the count and inflation portions of the model are statistically significant. This model fits the data significantly better than the null model, i.e., the intercept-only model. To show that this is the case, we can compare with the current model to a null model without predictors using chi-squared test on the difference of log likelihoods.

m0 <- **update**(m1, . ~ 1)

**pchisq**(2 \* (**logLik**(m1) - **logLik**(m0)), df = 3, lower.tail=FALSE)

## 'log Lik.' 1.28e-13 (df=6)

From the output above, we can see that our overall model is statistically significant.

Note that the model output above does not indicate in any way if our zero-inflated model is an improvement over a standard negative binomial regression. We can determine this by running the corresponding standard negative binomial model and then performing a Vuong test of the two models. We use the MASS package to run the standard negative binomial regression.

**summary**(m2 <- **glm.nb**(count ~ child + camper, data = zinb))

##

## Call:

## glm.nb(formula = count ~ child + camper, data = zinb, init.theta = 0.2552931119,

## link = log)

##

## Deviance Residuals:

## Min 1Q Median 3Q Max

## -1.314 -1.036 -0.727 -0.172 4.016

##

## Coefficients:

## Estimate Std. Error z value Pr(>|z|)

## (Intercept) 1.073 0.242 4.42 9.7e-06 \*\*\*

## child -1.375 0.196 -7.03 2.1e-12 \*\*\*

## camper1 0.909 0.284 3.21 0.0013 \*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## (Dispersion parameter for Negative Binomial(0.2553) family taken to be 1)

##

## Null deviance: 258.93 on 249 degrees of freedom

## Residual deviance: 201.89 on 247 degrees of freedom

## AIC: 887.4

##

## Number of Fisher Scoring iterations: 1

##

##

## Theta: 0.2553

## Std. Err.: 0.0329

##

## 2 x log-likelihood: -879.4210

**vuong**(m1, m2)

## Vuong Non-Nested Hypothesis Test-Statistic: 1.702

## (test-statistic is asymptotically distributed N(0,1) under the

## null that the models are indistinguishible)

## in this case:

## model1 > model2, with p-value 0.0444

* The predictors child and camper in the part of the negative binomial regression model predicting number of fish caught (count) are both significant predictors.
* The predictor person in the part of the logit model predicting excessive zeros is statistically significant.
* For these data, the expected change in log(count) for a one-unit increase in child is -1.515255 holding other variables constant.
* A camper (camper = 1) has an expected log(count) of 0.879051 higher than that of a non-camper (camper = 0) holding other variables constant.
* The log odds of being an excessive zero would decrease by 1.67 for every additional person in the group. In other words, the more people in the group the less likely that the zero would be due to not gone fishing. Put plainly, the larger the group the person was in, the more likely that the person went fishing.
* The Vuong test suggests that the zero-inflated negative binomial model is a significant improvement over a standard negative binomial model.

We can get confidence intervals for the parameters and the exponentiated parameters using bootstrapping. For the negative binomial model, these would be incident risk ratios, for the zero inflation model, odds ratios. We use the boot package. First, we get the coefficients from our original model to use as start values for the model to speed up the time it takes to estimate. Then we write a short function that takes data and indices as input and returns the parameters we are interested in. Finally, we pass that to the boot function and do 1200 replicates, using snow to distribute across four cores. Note that you should adjust the number of cores to whatever your machine has. Also, for final results, one may wish to increase the number of replications to help ensure stable results.

**dput**(**round**(**coef**(m1, "count"), 4))

## structure(c(1.3711, -1.5152, 0.879), .Names = c("(Intercept)",

## "child", "camper1"))

**dput**(**round**(**coef**(m1, "zero"), 4))

## structure(c(1.6028, -1.6663), .Names = c("(Intercept)", "persons"

## ))

f <- **function**(data, i) {

**require**(pscl)

m <- **zeroinfl**(count ~ child + camper | persons,

data = data[i, ], dist = "negbin",

start = **list**(count = **c**(1.3711, -1.5152, 0.879), zero = **c**(1.6028, -1.6663)))

**as.vector**(**t**(**do.call**(rbind, **coef**(**summary**(m)))[, 1:2]))

}

**set.seed**(10)

(res <- **boot**(zinb, f, R = 1200, parallel = "snow", ncpus = 4))

##

## ORDINARY NONPARAMETRIC BOOTSTRAP

##

##

## Call:

## boot(data = zinb, statistic = f, R = 1200, parallel = "snow",

## ncpus = 4)

##

##

## Bootstrap Statistics :

## original bias std. error

## t1\* 1.3711 -0.083023 0.39403

## t2\* 0.2561 -0.002622 0.03191

## t3\* -1.5153 -0.061487 0.26892

## t4\* 0.1956 0.006034 0.02027

## t5\* 0.8791 0.091431 0.47124

## t6\* 0.2693 0.001873 0.01998

## t7\* -0.9854 0.080120 0.21896

## t8\* 0.1760 0.002577 0.01689

## t9\* 1.6031 0.473597 1.59331

## t10\* 0.8365 3.767327 15.65780

## t11\* -1.6666 -0.462364 1.56789

## t12\* 0.6793 3.771994 15.69675

The results are alternating parameter estimates and standard errors. That is, the first row has the first parameter estimate from our model. The second has the standard error for the first parameter. The third column contains the bootstrapped standard errors, which are considerably larger than those estimated by zeroinfl.

Now we can get the confidence intervals for all the parameters. We start on the original scale with percentile and bias adjusted CIs. We also compare these results with the regular confidence intervals based on the standard errors.

*## basic parameter estimates with percentile and bias adjusted CIs*

parms <- **t**(**sapply**(**c**(1, 3, 5, 7, 9), **function**(i) {

out <- **boot.ci**(res, index = **c**(i, i + 1), type = **c**("perc", "bca"))

**with**(out, **c**(Est = t0, pLL = percent[4], pUL = percent[5],

bcaLL = bca[4], bcaLL = bca[5]))

}))

*## add row names*

**row.names**(parms) <- **names**(**coef**(m1))

*## print results*

parms

## Est pLL pUL bcaLL bcaLL

## count\_(Intercept) 1.3711 0.5132 2.0540 0.7577 2.3268

## count\_child -1.5153 -2.1495 -1.0807 -1.9872 -0.9827

## count\_camper1 0.8791 0.1119 1.8628 -0.2267 1.6648

## zero\_(Intercept) -0.9854 -1.3143 -0.4465 -1.4472 -0.6407

## zero\_persons 1.6031 0.3520 8.0796 -0.1858 3.6867

*## compare with normal based approximation*

**confint**(m1)

## 2.5 % 97.5 %

## count\_(Intercept) 0.86911 1.8731

## count\_child -1.89860 -1.1319

## count\_camper1 0.35127 1.4068

## zero\_(Intercept) -0.03636 3.2419

## zero\_persons -2.99701 -0.3355

The bootstrapped confidence intervals are considerably wider than the normal based approximation. The bootstrapped CIs are more consistent with the CIs from Stata when using robust standard errors.

Now we can estimate the incident risk ratio (IRR) for the negative binomial model and odds ratio (OR) for the logistic (zero inflation) model. This is done using almost identical code as before, but passing a transformation function to the h argument of boot.ci, in this case, exp to exponentiate.

*## exponentiated parameter estimates with percentile and bias adjusted CIs*

expparms <- **t**(**sapply**(**c**(1, 3, 5, 7, 9), **function**(i) {

out <- **boot.ci**(res, index = **c**(i, i + 1), type = **c**("perc", "bca"), h = exp)

**with**(out, **c**(Est = t0, pLL = percent[4], pUL = percent[5],

bcaLL = bca[4], bcaLL = bca[5]))

}))

*## add row names*

**row.names**(expparms) <- **names**(**coef**(m1))

*## print results*

expparms

## Est pLL pUL bcaLL bcaLL

## count\_(Intercept) 3.9395 1.6707 7.7990 2.1334 10.2453

## count\_child 0.2198 0.1165 0.3393 0.1371 0.3743

## count\_camper1 2.4086 1.1185 6.4420 0.7971 5.2845

## zero\_(Intercept) 0.3733 0.2687 0.6398 0.2352 0.5269

## zero\_persons 4.9686 1.4219 3227.9196 0.8305 39.9126

To better understand our model, we can compute the expected number of fish caught for different combinations of our predictors. In fact, since we are working with essentially categorical predictors, we can compute the expected values for all combinations using the expand.grid function to create all combinations and then the predict function to do it. Finally we create a graph.

newdata1 <- **expand.grid**(0:3, **factor**(0:1), 1:4)

**colnames**(newdata1) <- **c**("child", "camper", "persons")

newdata1$phat <- **predict**(m1, newdata1)

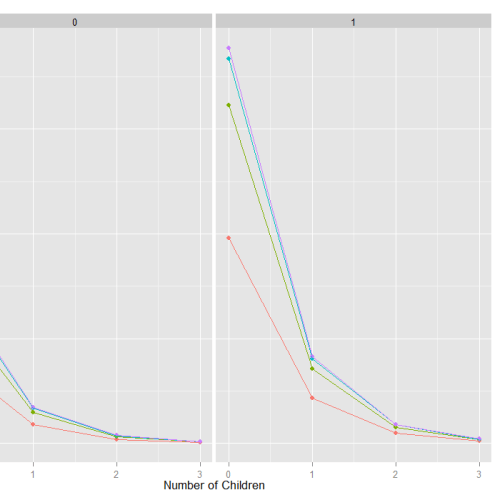
**ggplot**(newdata1, **aes**(x = child, y = phat, colour = **factor**(persons))) +

**geom\_point**() +

**geom\_line**() +

**facet\_wrap**(~camper) +

**labs**(x = "Number of Children", y = "Predicted Fish Caught")



Things to consider

Here are some issues that you may want to consider in the course of your research analysis.

* Question about the over-dispersion parameter is in general a tricky one. A large over-dispersion parameter could be due to a miss-specified model or could be due to a real process with over-dispersion. Adding an over-dispersion problem does not necessarily improve a miss-specified model.
* The zero inflated negative binomial model has two parts, a negative binomial count model and the logit model for predicting excess zeros, so you might want to review these Data Analysis Example pages, [Negative Binomial Regression](https://stats.idre.ucla.edu/r/dae/negative-binomial-regression/) and [Logit Regression](https://stats.idre.ucla.edu/r/dae/logit-regression/).
* Since zero inflated negative binomial has both a count model and a logit model, each of the two models should have good predictors. The two models do not necessarily need to use the same predictors.
* Problems of perfect prediction, separation or partial separation can occur in the logistic part of the zero-inflated model.
* Count data often use exposure variable to indicate the number of times the event could have happened. You can incorporate exposure (also called an offset) into your model by using the offset() function.
* It is not recommended that zero-inflated negative binomial models be applied to small samples. What constitutes a small sample does not seem to be clearly defined in the literature.
* Pseudo-R-squared values differ from OLS R-squareds, please see [FAQ: What are pseudo R-squareds?](https://stats.idre.ucla.edu/other/mult-pkg/faq/general/faq-what-are-pseudo-r-squareds/) for a discussion on this issue.

References

* Long, J. S. 1997. *Regression Models for Categorical and Limited Dependent Variables.* Thousand Oaks, CA: Sage Publications. Everitt, B. S. and Hothorn, T. [A Handbook of Statistical Analyses Using R](http://cran.r-project.org/web/packages/HSAUR/vignettes/preface.pdf)