432 Class 16 Slides

github.com/THOMASELOVE/2020-432

2020-03-24

Today's Topic

Regression Models for Count Outcomes

- Six modeling approaches are illustrated in these slides.
    - Poisson Regression
    - Negative Binomial Regression
    - Two types of Zero-inflated model
        - ZIP (Zero-inflated Poisson)
        - ZINB (Zero-inflated Neg. Binomial)
    - Two types of Hurdle model
        - using a Poisson approach
        - using a Negative Binomial approach

Chapter 19 of the Course Notes describes this material.

Setup

We've previously installed the countreg package from R-Forge.

```
library(magrittr); library(here); library(janitor)
library(knitr)
library(caret)
library(MASS)
library(pscl)
library(VGAM)
library(broom)
library(tidyverse)

theme_set(theme_bw())
```

An Overview

Generalized Linear Models for Count Outcomes

We want to build a generalized linear model to predict count data using one or more predictors.

In count data, the observations are non-negative integers (0, 1, 2, 3, . . .)

- the number of COVID-19 hospitalizations in Ohio yesterday
- the number of mutations within a particular search grid
- the number of days in the past 30 where your mental health was poor

The Poisson and the Negative Binomial probability distributions will be useful.

The Poisson Probability Distribution

The Poisson probability model describes the probability of a given number of events occurring in a fixed interval of time or space.

- If events occur with a constant mean rate, and independently of the time since the last event, the Poisson model is appropriate.
- The probability mass function for a discrete random variable with Poisson distribution follows.

$$Pr(Y = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

- k is the number of times an event occurs in an interval, and k can take the values 0, 1, 2, 3, ...
- The parameter λ (lambda) is equal to the expected value (mean) of Y and is also equal to the variance of Y.

The Negative Binomial Probability Distribution

The Negative Binomial distribution models the number of failures in a sequence of independent and identically distributed Bernoulli trials before a specified number of successes occurs.

- The probability mass function for a discrete random variable with a negative binomial distribution follows.

$$Pr(Y = k) = \binom{k + r - 1}{k} p^r (1 - p)^k$$

- k is the number of failures (units of time) before the rth event occurs, and k can take the values 0, 1, 2, 3, . . .
- The mean of the random variable Y which follows a negative binomial distribution is $rp/(1 - p)$ and the variance is $rp/(1 - p)^2$.

Poisson Regression and the possibility of overdispersion

- Poisson regression assumes that the outcome Y follows a Poisson distribution, and that the logarithm of the expected value of Y (its mean) can be modeled by a linear combination of a set of predictors.
    - A Poisson regression makes the strong assumption that the variance of Y is equal to its mean.
    - A Poisson model might fit poorly due to **overdispersion**, where the variance of Y is larger than we'd expect based on the mean of Y.
    - Quasipoisson models are available which estimate an overdispersion parameter, but we'll skip those.

We will show the use of `glm` to fit Poisson models, by using `family = "Poisson"`.

Negative Binomial Regression to generalize the Poisson

- Negative binomial regression is a generalization of Poisson regression which loosens the assumption that the variance of Y is equal to its mean, and thus produces models which fit a broader class of data.

We will demonstrate the use of `glm.nb` from the `MASS` package to fit negative binomial regression models.

Zero-inflated approaches

- Both the Poisson and Negative Binomial regression approaches may under-estimate the number of zeros compared to the data.
- To better match up the counts of zero, zero-inflated models fit:
    - a logistic regression to predict the extra zeros, along with
    - a Poisson or Negative Binomial model to predict the counts, including some zeros.

We will demonstrate the use of `zeroinfl` from the `pscl` package to fit zero-inflated Poisson (or ZIP) and zero-inflated negative binomial (or ZINB) regressions.

Hurdle models

A hurdle model predicts the count outcome by making an assumption that there are two processes at work:

- a process that determines whether the count is zero or not zero (usually using logistic regression), and
- a process that determines the count when we know the subject has a positive count (usually using a truncated Poisson or Negative Binomial model where no zeros are predicted)

We'll use the `hurdle` function from the `pscl` package to fit these models.

Comparing Models

1. A key tool will be a graphical representation of the fit of the models to the count outcome, called a **rootogram**. We'll use the rootograms produced by the `countreg` package to help us.
2. We'll also demonstrate a Vuong hypothesis testing approach (from the `lmtest` package) to help us make decisions between various types of Poisson models or various types of Negative Binomial models on the basis of improvement in fit of things like bias-corrected AIC or BIC.
3. We'll also demonstrate the calculation of pseudo-R square statistics for comparing models, which can be compared in a validation sample as well as in the original modeling sample.

The `medicare` data

The medicare example

The data we will use come from the NMES1988 data set in R's AER package, although I have built a cleaner version for you in the medicare.csv file on our web site. These are essentially the same data as are used in my main resource from the University of Virginia for hurdle models.

These data are a cross-section originating from the US National Medical Expenditure Survey (NMES) conducted in 1987 and 1988. The NMES is based upon a representative, national probability sample of the civilian non-institutionalized population and individuals admitted to long-term care facilities during 1987. The data are a subsample of individuals ages 66 and over all of whom are covered by Medicare (a public insurance program providing substantial protection against health-care costs), and some of whom also have private supplemental insurance.

```
medicare <- read.csv(here("data/medicare.csv")) %>% tbl_df
```

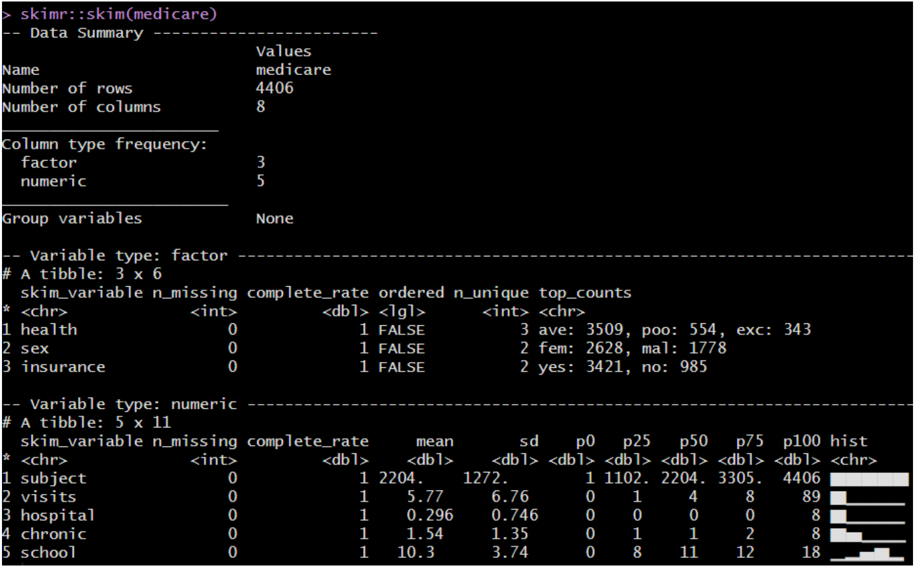
The medicare code book

Variable	Description
subject	subject number (code)
visits	outcome of interest: number of physician office visits
hospital	number of hospital stays
health	self-perceived health status (poor, average, excellent)
chronic	number of chronic conditions
sex	male or female
school	number of years of education
insurance	is the subject (also) covered by private insurance? (yes or no)

Today's Goal

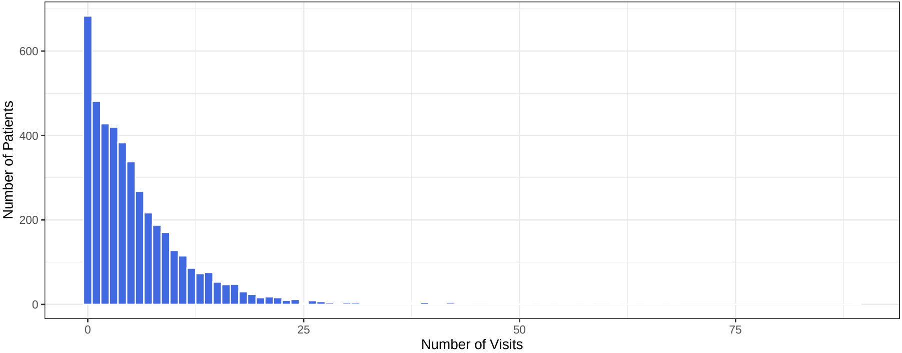
Predict visits using main effects of the 6 predictors (excluding subject)

Skimming the `medicare` tibble



```
> skimr::skim(medicare)
-- Data Summary ------------------------
                           Values
Name                       medicare
Number of rows             4406
Number of columns          8
_______________________
Column type frequency:
  factor                   3
  numeric                  5
________________________
Group variables            None

-- Variable type: factor ------------------------------------------------------------------
# A tibble: 3 x 6
  skim_variable n_missing complete_rate ordered n_unique top_counts
* <chr>             <int>         <dbl> <lgl>      <int> <chr>
1 health                0             1 FALSE          3 ave: 3509, poo: 554, exc: 343
2 sex                   0             1 FALSE          2 fem: 2628, mal: 1778
3 insurance             0             1 FALSE          2 yes: 3421, no: 985

-- Variable type: numeric -----------------------------------------------------------------
# A tibble: 5 x 11
  skim_variable n_missing complete_rate    mean     sd    p0   p25   p50   p75  p100 hist
* <chr>             <int>         <dbl>   <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <chr>
1 subject               0             1 2204.   1272.      1 1102. 2204. 3305.  4406
2 visits                0             1    5.77    6.76    0     1     4     8    89
3 hospital              0             1    0.296   0.746   0     0     0     0     8
4 chronic               0             1    1.54    1.35    0     1     1     2     8
5 school                0             1   10.3     3.74    0     8    11    12    18
```

Our outcome, `visits`



```
mosaic::favstats(~ visits, data = medicare)
```

```
 min Q1 median Q3 max     mean       sd    n missing
   0  1      4  8  89 5.774399 6.759225 4406       0
```

visits numerical summaries

```
medicare %$% Hmisc::describe(visits)
```

```
visits
       n  missing distinct     Info     Mean      Gmd
    4406        0       60    0.992    5.774    6.227
     .05      .10      .25      .50      .75      .90
       0        0        1        4        8       13
     .95
      17

lowest :  0  1  2  3  4, highest: 63 65 66 68 89
```

Reiterating the Goal

Predict `visits` using some combination of these 6 predictors...

Predictor	Description
`hospital`	number of hospital stays
`health`	self-perceived health status (poor, average, excellent)
`chronic`	number of chronic conditions
`sex`	male or female
`school`	number of years of education
`insurance`	is the subject (also) covered by private insurance? (yes or no)

We'll build separate training and test samples to help us validate.

Partitioning the Data into Training vs. Test Samples

```r
set.seed(432)
validation_samples <- medicare$visits %>%
  createDataPartition(p = 0.75, list = FALSE)

med_train = medicare[validation_samples,]
med_test = medicare[-validation_samples,]
```

I've held out 25% of the `medicare` data for the test sample.

```r
dim(med_train)
```

```
[1] 3306    8
```

```r
dim(med_test)
```

```
[1] 1100    8
```

`mod_1`: A Poisson Regression

Poisson Regression

Assume our count data (visits) follows a Poisson distribution with a mean conditional on our predictors.

```r
mod_1 <- glm(visits ~ hospital + health + chronic +
                  sex + school + insurance,
              data = med_train, family = "poisson")
```

The Poisson model uses a logarithm as its link function, so the model is actually predicting log(visits).

Note that we're fitting the model here using the training sample alone.

`mod_1` (Poisson) model coefficients

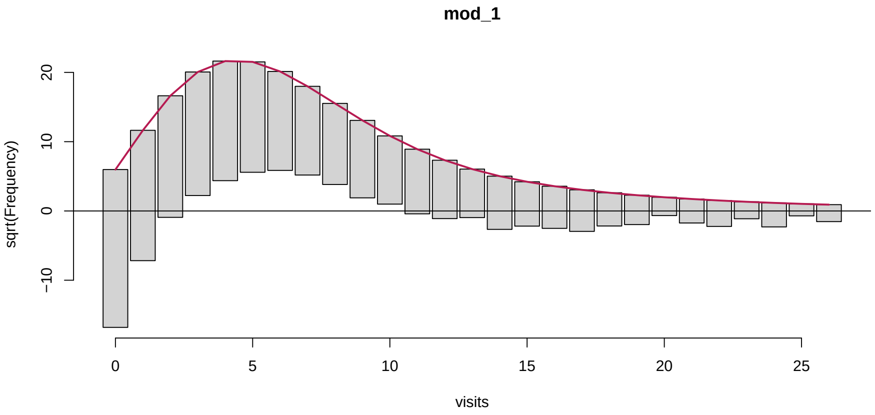
```r
tidy(mod_1) %>% kable(digits = c(0, 3, 3, 1, 3))
```

term	estimate	std.error	statistic	p.value
(Intercept)	0.990	0.028	35.8	0
hospital	0.165	0.007	24.7	0
healthexcellent	-0.384	0.035	-11.0	0
healthpoor	0.290	0.021	14.1	0
chronic	0.143	0.005	27.2	0
sexmale	-0.085	0.015	-5.7	0
school	0.032	0.002	15.2	0
insuranceyes	0.153	0.019	7.9	0

If Harry and Larry have the same values for all other predictors but only Harry has private insurance, the model predicts Harry to have a 0.153 point larger value of log(`visits`) than Larry.

Visualize fit with a (Hanging) Rootogram

```
countreg::rootogram(mod_1)
```



See the next slide for details on how to interpret this...

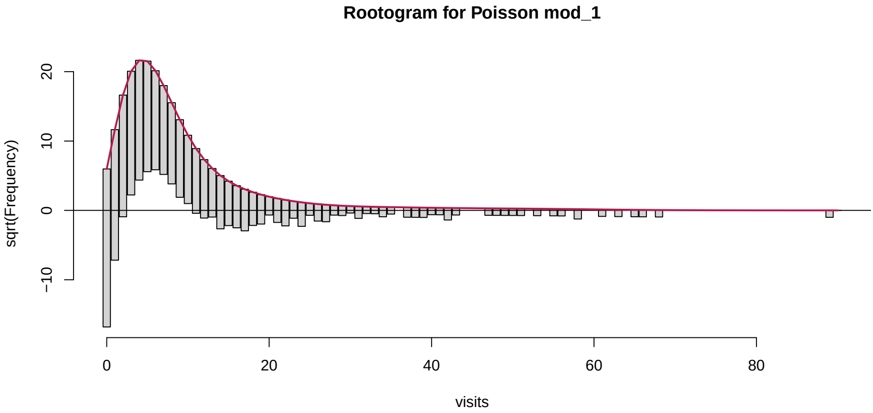
Interpreting the Rootogram

- The red curved line is the theoretical Poisson fit.
- "Hanging" from each point on the red line is a bar, the height of which represents the observed counts.
    - A bar hanging below 0 indicates that the model under-predicts that value. (Model predicts fewer values than the data show.)
    - A bar hanging above 0 indicates over-prediction of that value. (Model predicts more values than the data show.)
- The counts have been transformed with a square root transformation to prevent smaller counts from getting obscured and overwhelmed by larger counts.

For more information on rootograms, check out https://arxiv.org/pdf/1605.01311.

The Complete Rootogram for `mod_1`

```
countreg::rootogram(mod_1, max = 90,
                    main = "Rootogram for Poisson mod_1")
```



This shows what happens with the subject with 89 visits.

Interpreting the Rootogram for `mod_1`

In `mod_1`, we see a great deal of underfitting for counts of 0 and 1, then overfitting for visit counts in the 3-10 range, with some underfitting again at more than a dozen or so visits.

- Our Poisson model (`mod_1`) doesn't fit enough zeros or ones, and fits too many 3-12 values, then not enough of the higher values.

Store Training Sample `mod_1` Predictions

We'll use the `augment` function to store the predictions within our training sample. Note the use of `"response"` to predict `visits`, not log(`visits`).

```
mod_1_aug <- augment(mod_1, med_train,
                     type.predict = "response",
                     type.residuals = "response")

mod_1_aug %>% select(subject, visits, .fitted, .resid) %>%
  head(3)
```

```
# A tibble: 3 x 4
  subject visits .fitted .resid
    <int>  <int>   <dbl>  <dbl>
1       1      5    5.49 -0.492
2       2      1    5.77 -4.77
3       3     13   14.5  -1.45
```

Summarizing Training Sample mod_1 Fit

Within our training sample, mod_1_aug now contains both the actual counts (visits) and the predicted counts (in .fitted) from mod_1. We'll summarize the fit. . .

```
mod_1_summary <- tibble(
  model = "mod_1 (Poisson)",
  R2 = R2(mod_1_aug$.fitted, mod_1_aug$visits),
  RMSE = RMSE(mod_1_aug$.fitted, mod_1_aug$visits),
  MAE = MAE(mod_1_aug$.fitted, mod_1_aug$visits))

mod_1_summary %>% kable(digits = 3)
```

model	R2	RMSE	MAE
mod_1 (Poisson)	0.102	6.522	4.124

These will become interesting as we build additional models.

mod_2: A Negative Binomial Regression

Fitting the Negative Binomial Model

The negative binomial model requires the estimation of an additional parameter, called θ (theta). The default link for this generalized linear model is also a logarithm, like the Poisson.

```
mod_2 <- MASS::glm.nb(visits ~ hospital + health + chronic +
                         sex + school + insurance,
                      data = med_train)
```

The estimated dispersion parameter value θ is...

```
summary(mod_2)$theta
```

```
[1] 1.212527
```

The Poisson model is essentially the negative binomial model assuming a known $\theta = 1$.

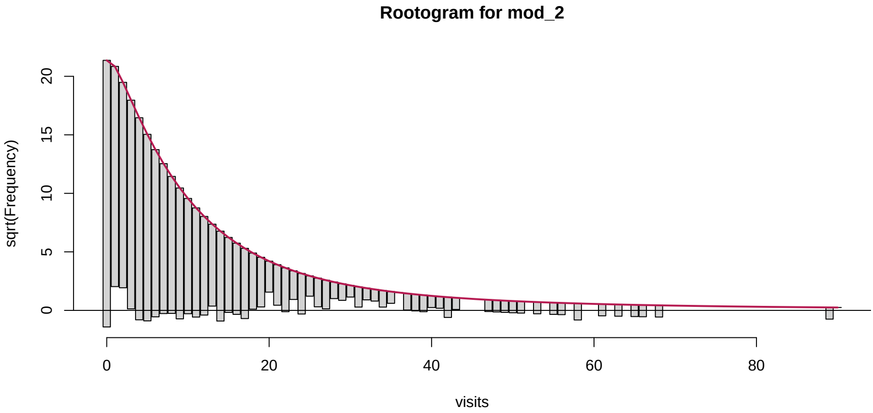
mod_2 (Negative Binomial) coefficients

```
tidy(mod_2) %>% kable(digits = c(0, 3, 3, 1, 3))
```

term	estimate	std.error	statistic	p.value
(Intercept)	0.873	0.064	13.7	0.000
hospital	0.220	0.023	9.6	0.000
healthexcellent	-0.369	0.070	-5.3	0.000
healthpoor	0.341	0.056	6.1	0.000
chronic	0.176	0.014	12.7	0.000
sexmale	-0.103	0.036	-2.9	0.004
school	0.033	0.005	6.4	0.000
insuranceyes	0.196	0.046	4.3	0.000

Rootogram for Negative Binomial Model

```
countreg::rootogram(mod_2, max = 90,
                    main = "Rootogram for mod_2")
```



Does this look better than the Poisson rootogram?

Store Training Sample mod_2 Predictions

```
mod_2_aug <- augment(mod_2, med_train,
                     type.predict = "response",
                     type.residuals = "response")

mod_2_aug %>% select(subject, visits, .fitted, .resid) %>%
  head(3)
```

```
# A tibble: 3 x 4
  subject visits .fitted .resid
    <int>  <int>   <dbl>  <dbl>
1       1      5    5.66 -0.659
2       2      1    5.73 -4.73
3       3     13   18.2  -5.23
```

Summarizing Training Sample `mod_2` Fit

As before, `mod_2_aug` now has actual (`visits`) and predicted counts (in `.fitted`) from `mod_2`.

```r
mod_2_summary <- tibble(
  model = "mod_2 (Neg. Binomial)",
  R2 = R2(mod_2_aug$.fitted, mod_2_aug$visits),
  RMSE = RMSE(mod_2_aug$.fitted, mod_2_aug$visits),
  MAE = MAE(mod_2_aug$.fitted, mod_2_aug$visits))

mod_2_summary %>% kable(digits = 3)
```

model	R2	RMSE	MAE
mod_2 (Neg. Binomial)	0.084	6.834	4.179

So Far in our Training Sample

The reasonable things to summarize in sample look like the impressions from the rootograms and the summaries we've prepared so far.

```
bind_rows(mod_1_summary, mod_2_summary) %>%
  kable(digits = 3)
```

model	R2	RMSE	MAE
mod_1 (Poisson)	0.102	6.522	4.124
mod_2 (Neg. Binomial)	0.084	6.834	4.179

Model	Rootogram impressions
`mod_1`	Many problems. Data appear overdispersed.
`mod_2`	Still not enough zeros; some big predictions.

`mod_3`: Zero-Inflated Poisson (ZIP) Model

Zero-Inflated Poisson (ZIP) model

The zero-inflated Poisson model describes count data with an excess of zero counts.

The model posits that there are two processes involved:

- a logistic regression model is used to predict excess zeros
- while a Poisson model is used to predict the counts

We'll use the pscl package to fit zero-inflated models.

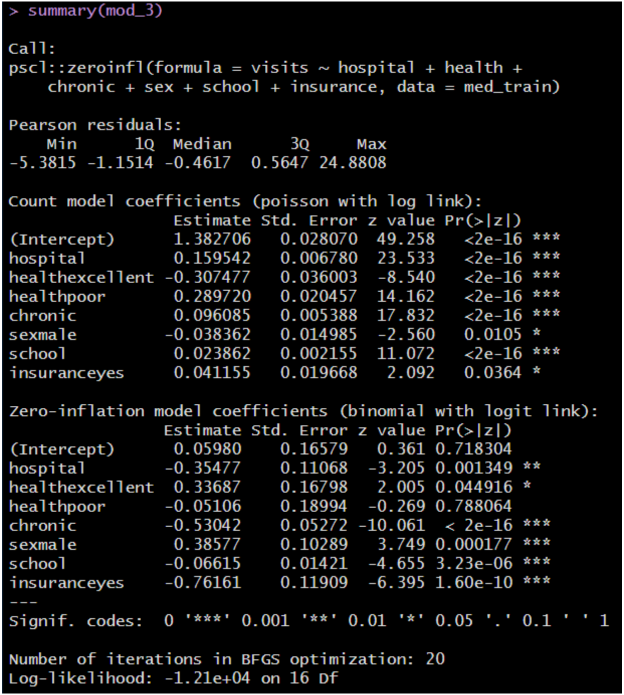
```
mod_3 <- pscl::zeroinfl(visits ~ hospital + health +
                    chronic + sex + school + insurance,
                    data = med_train)
```

mod_3 ZIP coefficients

Sadly, there's no broom tidying functions for these zero-inflated models.

```r
summary(mod_3)
```

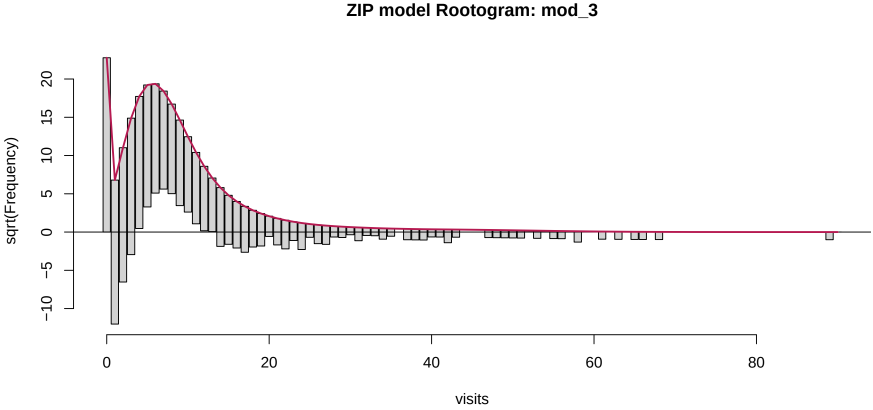
Screenshot on next slide. . .



```
> summary(mod_3)

Call:
pscl::zeroinfl(formula = visits ~ hospital + health + 
    chronic + sex + school + insurance, data = med_train)

Pearson residuals:
    Min      1Q  Median      3Q     Max 
-5.3815 -1.1514 -0.4617  0.5647 24.8808 

Count model coefficients (poisson with log link):
                 Estimate Std. Error z value Pr(>|z|)    
(Intercept)      1.382706   0.028070  49.258   <2e-16 ***
hospital         0.159542   0.006780  23.533   <2e-16 ***
healthexcellent -0.307477   0.036003  -8.540   <2e-16 ***
healthpoor       0.289720   0.020457  14.162   <2e-16 ***
chronic          0.096085   0.005388  17.832   <2e-16 ***
sexmale         -0.038362   0.014985  -2.560   0.0105 *  
school           0.023862   0.002155  11.072   <2e-16 ***
insuranceyes     0.041155   0.019668   2.092   0.0364 *  

Zero-inflation model coefficients (binomial with logit link):
                Estimate Std. Error z value Pr(>|z|)    
(Intercept)      0.05980    0.16579   0.361 0.718304    
hospital        -0.35477    0.11068  -3.205 0.001349 ** 
healthexcellent  0.33687    0.16798   2.005 0.044916 *  
healthpoor      -0.05106    0.18994  -0.269 0.788064    
chronic         -0.53042    0.05272 -10.061  < 2e-16 ***
sexmale          0.38577    0.10289   3.749 0.000177 ***
school          -0.06615    0.01421  -4.655 3.23e-06 ***
insuranceyes    -0.76161    0.11909  -6.395 1.60e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Number of iterations in BFGS optimization: 20 
Log-likelihood: -1.21e+04 on 16 Df
```

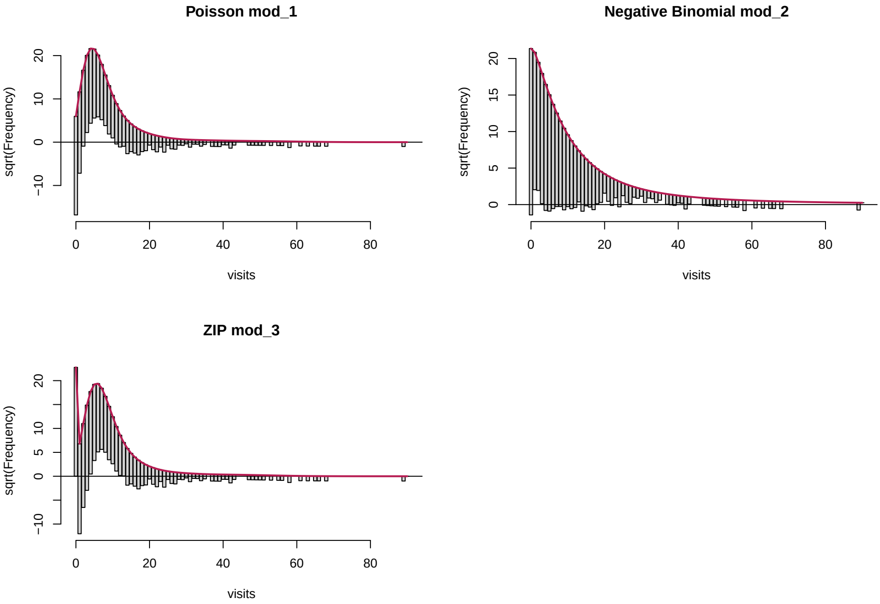
Rootogram for ZIP model

```
countreg::rootogram(mod_3, max = 90,
                    main = "ZIP model Rootogram: mod_3")
```



What do you think? Next slide shows all models so far.

First Three Rootograms - Which Looks Best?



Store Training Sample mod_3 Predictions

We have no augment or other broom functions available for zero-inflated models, so . . .

```
mod_3_aug <- med_train %>%
    mutate(".fitted" = predict(mod_3, type = "response"),
           ".resid" = resid(mod_3, type = "response"))

mod_3_aug %>% select(subject, visits, .fitted, .resid) %>%
  head(3)
```

```
# A tibble: 3 x 4
  subject visits .fitted .resid
    <int>  <int>   <dbl>  <dbl>
1       1      5    5.86 -0.859
2       2      1    5.87 -4.87
3       3     13   15.7  -2.69
```

Summarizing Training Sample `mod_3` Fit

`mod_3_aug` now has actual (`visits`) and predicted counts (in `.fitted`) from `mod_3`, just as we set up for the previous two models.

```r
mod_3_summary <- tibble(
  model = "mod_3 (ZIP)",
  R2 = R2(mod_3_aug$.fitted, mod_3_aug$visits),
  RMSE = RMSE(mod_3_aug$.fitted, mod_3_aug$visits),
  MAE = MAE(mod_3_aug$.fitted, mod_3_aug$visits))

mod_3_summary %>% kable(digits = 3)
```

model	R2	RMSE	MAE
mod_3 (ZIP)	0.113	6.481	4.093

Training Sample Results through `mod_3`

```
bind_rows(mod_1_summary, mod_2_summary, mod_3_summary) %>%
  kable(digits = 3)
```

model	R2	RMSE	MAE
mod_1 (Poisson)	0.102	6.522	4.124
mod_2 (Neg. Binomial)	0.084	6.834	4.179
mod_3 (ZIP)	0.113	6.481	4.093

Remember we want a larger R^2 and smaller values of RMSE and MAE.

Comparing models with Vuong's procedure

Vuong's test compares predicted probabilities (for each count) in two non-nested models. How about Poisson vs. ZIP?

```r
vuong(mod_1, mod_3)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A    p-value
Raw                   -14.93727 model2 > model1 < 2.22e-16
AIC-corrected         -14.85194 model2 > model1 < 2.22e-16
BIC-corrected         -14.59155 model2 > model1 < 2.22e-16
```

The large negative z-statistic indicates mod_3 (ZIP) fits detectably better than mod_1 (Poisson) in our training sample.

Reference: Vuong, QH (1989) Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica*, 57:307-333.

`mod_4`: Zero-Inflated Negative Binomial (ZINB) Model

Zero-Inflated Negative Binomial (ZINB) model

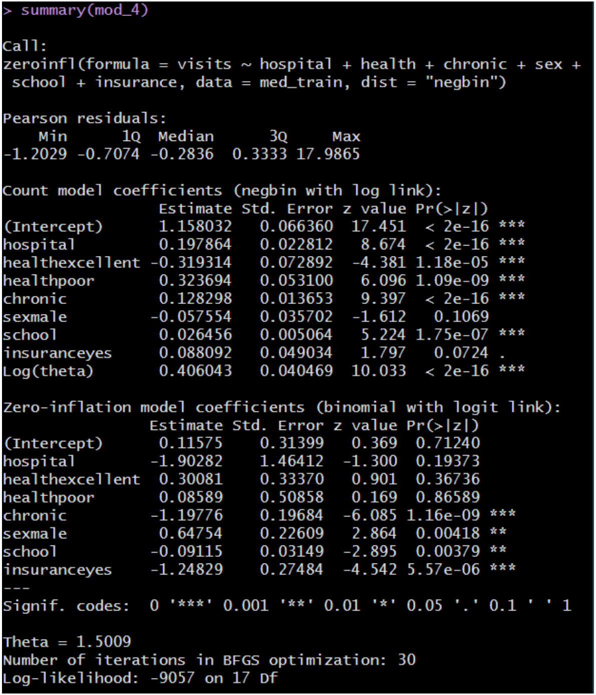
As in the ZIP, we assume there are two processes involved:

- a logistic regression model is used to predict excess zeros
- while a negative binomial model is used to predict the counts

We'll use the `pscl` package again and the `zeroinfl` function.

```
mod_4 <- zeroinfl(visits ~ hospital + health + chronic +
                      sex + school + insurance,
                  dist = "negbin", data = med_train)
```

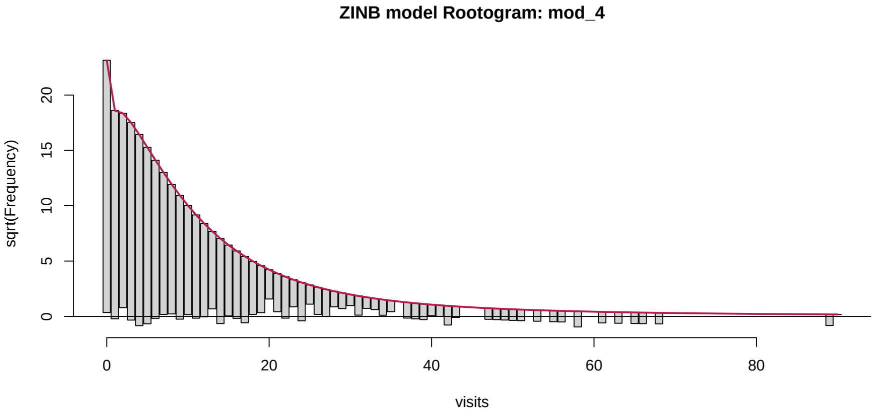
`summary(mod_4)` results on next slide. . .



```
> summary(mod_4)

Call:
zeroinfl(formula = visits ~ hospital + health + chronic + sex +
 school + insurance, data = med_train, dist = "negbin")

Pearson residuals:
    Min      1Q  Median      3Q     Max
-1.2029 -0.7074 -0.2836  0.3333 17.9865

Count model coefficients (negbin with log link):
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      1.158032   0.066360  17.451  < 2e-16 ***
hospital         0.197864   0.022812   8.674  < 2e-16 ***
healthexcellent -0.319314   0.072892  -4.381 1.18e-05 ***
healthpoor       0.323694   0.053100   6.096 1.09e-09 ***
chronic          0.128298   0.013653   9.397  < 2e-16 ***
sexmale         -0.057554   0.035702  -1.612   0.1069
school           0.026456   0.005064   5.224 1.75e-07 ***
insuranceyes     0.088092   0.049034   1.797   0.0724 .
Log(theta)       0.406043   0.040469  10.033  < 2e-16 ***

Zero-inflation model coefficients (binomial with logit link):
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      0.11575    0.31399   0.369  0.71240
hospital        -1.90282    1.46412  -1.300  0.19373
healthexcellent  0.30081    0.33370   0.901  0.36736
healthpoor       0.08589    0.50858   0.169  0.86589
chronic         -1.19776    0.19684  -6.085 1.16e-09 ***
sexmale          0.64754    0.22609   2.864  0.00418 **
school          -0.09115    0.03149  -2.895  0.00379 **
insuranceyes    -1.24829    0.27484  -4.542 5.57e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Theta = 1.5009
Number of iterations in BFGS optimization: 30
Log-likelihood: -9057 on 17 Df
```

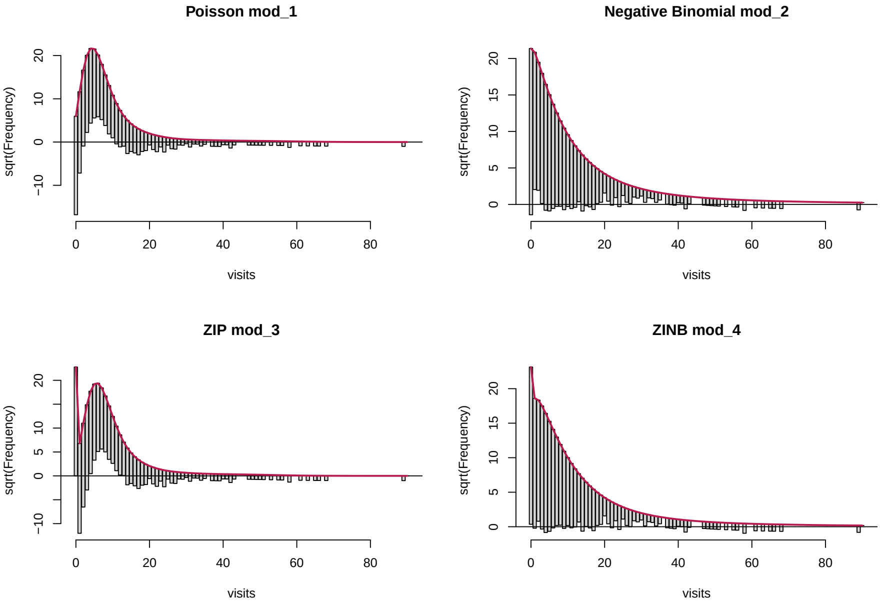
Rootogram for ZIP model

```
countreg::rootogram(mod_4, max = 90,
                    main = "ZINB model Rootogram: mod_4")
```



Again, next slide shows all models so far.

First Four Rootograms - Which Looks Best?



Store Training Sample `mod_4` Predictions

Again, there is no `augment` or other `broom` functions available for zero-inflated models, so . . .

```
mod_4_aug <- med_train %>%
    mutate(".fitted" = predict(mod_4, type = "response"),
           ".resid" = resid(mod_4, type = "response"))

mod_4_aug %>% select(subject, visits, .fitted, .resid) %>%
  head(3)
```

```
# A tibble: 3 x 4
  subject visits .fitted .resid
    <int>  <int>   <dbl>  <dbl>
1       1      5    6.03  -1.03
2       2      1    5.79  -4.79
3       3     13   17.3   -4.34
```

Summarizing Training Sample `mod_4` Fit

`mod_4_aug` now has actual (`visits`) and predicted counts (in `.fitted`) from `mod_4`.

```r
mod_4_summary <- tibble(
  model = "mod_4 (ZINB)",
  R2 = R2(mod_4_aug$.fitted, mod_4_aug$visits),
  RMSE = RMSE(mod_4_aug$.fitted, mod_4_aug$visits),
  MAE = MAE(mod_4_aug$.fitted, mod_4_aug$visits))

mod_4_summary %>% kable(digits = 3)
```

model	R2	RMSE	MAE
mod_4 (ZINB)	0.101	6.592	4.111

Training Sample Results through `mod_4`

```
bind_rows(mod_1_summary, mod_2_summary,
          mod_3_summary, mod_4_summary) %>%
  kable(digits = 3)
```

model	R2	RMSE	MAE
mod_1 (Poisson)	0.102	6.522	4.124
mod_2 (Neg. Binomial)	0.084	6.834	4.179
mod_3 (ZIP)	0.113	6.481	4.093
mod_4 (ZINB)	0.101	6.592	4.111

What do you think?

Comparing models with Vuong's procedure

Vuong's test compares predicted probabilities (for each count) in two non-nested models. How about Negative Binomial vs. ZINB?

```r
vuong(mod_4, mod_2)
```

```
Vuong Non-Nested Hypothesis Test-Statistic: 
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A    p-value
Raw                    5.414748 model1 > model2 3.0688e-08
AIC-corrected          4.777133 model1 > model2 8.8906e-07
BIC-corrected          2.831291 model1 > model2   0.002318
```

The large positive z-statistics indicate `mod_4` (ZINB) fits detectably better than `mod_2` (Negative Binomial) in our training sample.

`mod_5`: Poisson-Logistic Hurdle Model

The Hurdle Model

The hurdle model is a two-part model that specifies one process for zero counts and another process for positive counts. The idea is that positive counts occur once a threshold is crossed, or put another way, a hurdle is cleared. If the hurdle is not cleared, then we have a count of 0.

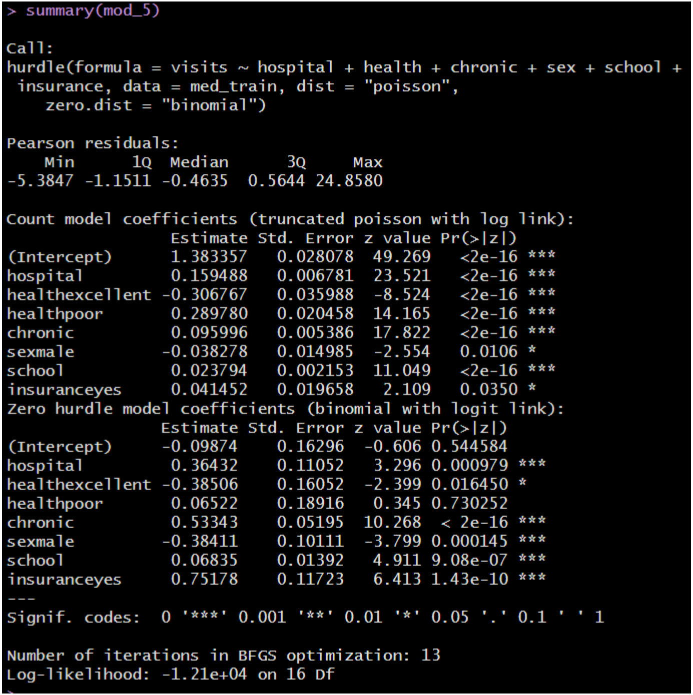
- The first part of the model is typically a **binary logistic regression** model. This models whether an observation takes a positive count or not.
- The second part of the model is usually a truncated Poisson or Negative Binomial model. Truncated means we're only fitting positive counts, and not zeros.

Fitting a Hurdle Model / Poisson-Logistic

In fitting a hurdle model to our medicare training data, the interpretation would be that one process governs whether a patient visits a doctor or not, and another process governs how many visits are made.

```
mod_5 <- hurdle(visits ~ hospital + health + chronic +
                  sex + school + insurance,
              dist = "poisson", zero.dist = "binomial",
              data = med_train)
```

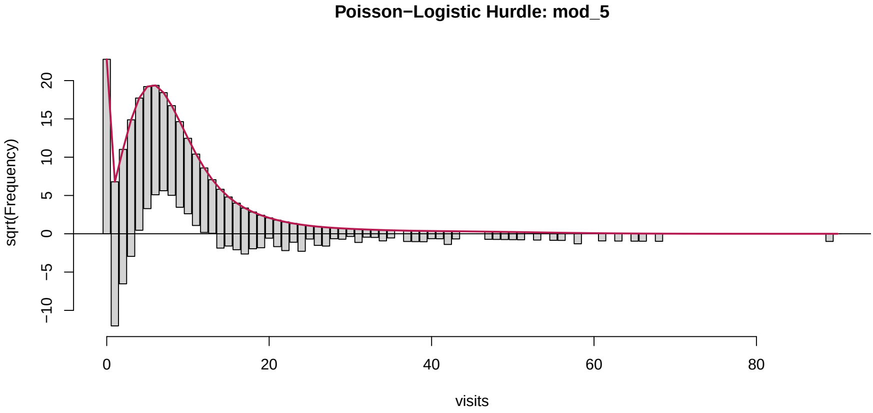
`summary(mod_5)` results follow. . .



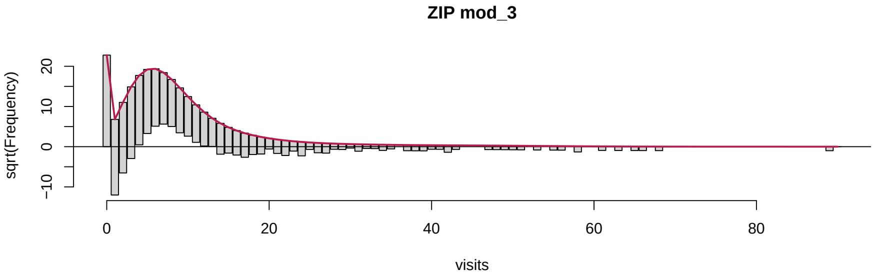
```
> summary(mod_5)

Call:
hurdle(formula = visits ~ hospital + health + chronic + sex + school +
 insurance, data = med_train, dist = "poisson",
    zero.dist = "binomial")

Pearson residuals:
    Min      1Q  Median      3Q     Max
-5.3847 -1.1511 -0.4635  0.5644 24.8580

Count model coefficients (truncated poisson with log link):
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      1.383357   0.028078  49.269   <2e-16 ***
hospital         0.159488   0.006781  23.521   <2e-16 ***
healthexcellent -0.306767   0.035988  -8.524   <2e-16 ***
healthpoor       0.289780   0.020458  14.165   <2e-16 ***
chronic          0.095996   0.005386  17.822   <2e-16 ***
sexmale         -0.038278   0.014985  -2.554   0.0106 *
school           0.023794   0.002153  11.049   <2e-16 ***
insuranceyes     0.041452   0.019658   2.109   0.0350 *
Zero hurdle model coefficients (binomial with logit link):
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -0.09874    0.16296  -0.606 0.544584
hospital         0.36432    0.11052   3.296 0.000979 ***
healthexcellent -0.38506    0.16052  -2.399 0.016450 *
healthpoor       0.06522    0.18916   0.345 0.730252
chronic          0.53343    0.05195  10.268  < 2e-16 ***
sexmale         -0.38411    0.10111  -3.799 0.000145 ***
school           0.06835    0.01392   4.911 9.08e-07 ***
insuranceyes     0.75178    0.11723   6.413 1.43e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Number of iterations in BFGS optimization: 13
Log-likelihood: -1.21e+04 on 16 Df
```

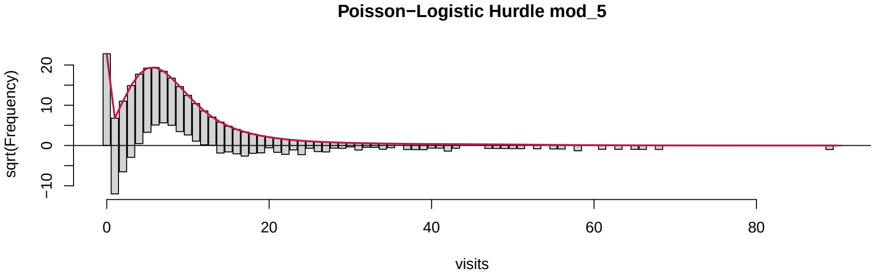
Rootogram for Poisson-Logistic Hurdle model

```
countreg::rootogram(mod_5, max = 90,
                    main = "Poisson-Logistic Hurdle: mod_5")
```



Poisson-Based Rootograms - Which Looks Best?





Store Training Sample `mod_5` Predictions

No augment or other broom functions for hurdle models, so ...

```
mod_5_aug <- med_train %>%
    mutate(".fitted" = predict(mod_5, type = "response"),
           ".resid" = resid(mod_5, type = "response"))

mod_5_aug %>% select(subject, visits, .fitted, .resid) %>%
  head(3)
```

```
# A tibble: 3 x 4
  subject visits .fitted .resid
    <int>  <int>   <dbl>  <dbl>
1       1      5    5.86 -0.858
2       2      1    5.87 -4.87
3       3     13   15.7  -2.69
```

Summarizing Training Sample mod_5 Fit

mod_5_aug has actual (visits) and mod_5 predicted (in .fitted) counts.

```
mod_5_summary <- tibble(
  model = "mod_5 (Poisson Hurdle)",
  R2 = R2(mod_5_aug$.fitted, mod_5_aug$visits),
  RMSE = RMSE(mod_5_aug$.fitted, mod_5_aug$visits),
  MAE = MAE(mod_5_aug$.fitted, mod_5_aug$visits))

mod_5_summary %>% kable(digits = 3)
```

model	R2	RMSE	MAE
mod_5 (Poisson Hurdle)	0.113	6.481	4.093

Training Sample Results through `mod_5`

```r
bind_rows(mod_1_summary, mod_2_summary,
          mod_3_summary, mod_4_summary,
          mod_5_summary) %>%
  kable(digits = 3)
```

model	R2	RMSE	MAE
mod_1 (Poisson)	0.102	6.522	4.124
mod_2 (Neg. Binomial)	0.084	6.834	4.179
mod_3 (ZIP)	0.113	6.481	4.093
mod_4 (ZINB)	0.101	6.592	4.111
mod_5 (Poisson Hurdle)	0.113	6.481	4.093

What do you think?

Are ZIP and Poisson-Logistic Hurdle the Same?

```
temp_check <- tibble(
  subject = mod_3_aug$subject,
  visits = mod_3_aug$visits,
  pred_zip = mod_3_aug$.fitted,
  pred_hur = mod_5_aug$.fitted,
  diff = pred_hur - pred_zip)

mosaic::favstats(~ diff, data = temp_check)
```

```
        min            Q1       median           Q3
-0.02810685 -0.0005787973 0.0002147671 0.0008459066
       max         mean          sd    n missing
0.04125959 0.0003326582 0.003323227 3306       0
```

Vuong test: Comparing mod_3 and mod_5

```
vuong(mod_3, mod_5)
```

```
Vuong Non-Nested Hypothesis Test-Statistic: 
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A p-value
Raw                    1.950967 model1 > model2 0.02553
AIC-corrected          1.950967 model1 > model2 0.02553
BIC-corrected          1.950967 model1 > model2 0.02553
```

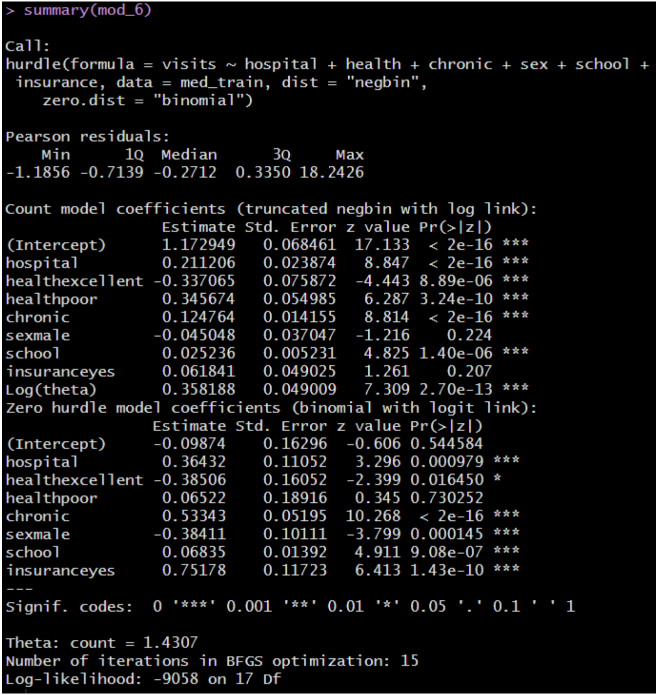
There's some evidence mod_3 (ZIP) fits better than mod_5 (Hurdle) in our training sample.

`mod_6`: Negative Binomial-Logistic Hurdle Model

Fitting a Hurdle Model / NB-Logistic

```r
mod_6 <- hurdle(visits ~ hospital + health + chronic +
                    sex + school + insurance,
                dist = "negbin", zero.dist = "binomial",
                data = med_train)
```

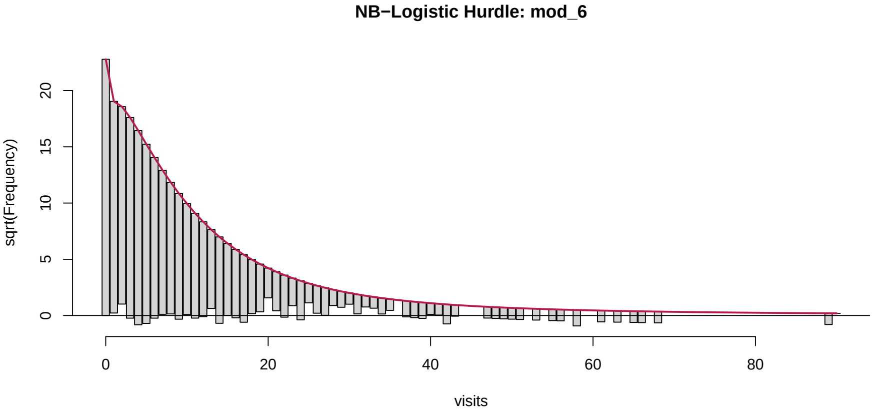
`summary(mod_6)` results follow. . .



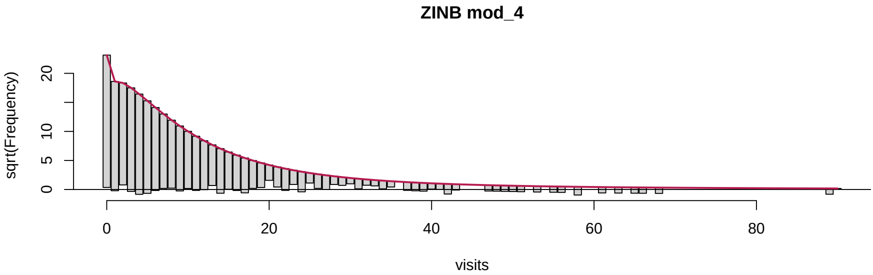
```
> summary(mod_6)

Call:
hurdle(formula = visits ~ hospital + health + chronic + sex + school +
 insurance, data = med_train, dist = "negbin",
    zero.dist = "binomial")

Pearson residuals:
    Min      1Q  Median      3Q     Max
-1.1856 -0.7139 -0.2712  0.3350 18.2426

Count model coefficients (truncated negbin with log link):
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      1.172949   0.068461  17.133  < 2e-16 ***
hospital         0.211206   0.023874   8.847  < 2e-16 ***
healthexcellent -0.337065   0.075872  -4.443 8.89e-06 ***
healthpoor       0.345674   0.054985   6.287 3.24e-10 ***
chronic          0.124764   0.014155   8.814  < 2e-16 ***
sexmale         -0.045048   0.037047  -1.216    0.224
school           0.025236   0.005231   4.825 1.40e-06 ***
insuranceyes     0.061841   0.049025   1.261    0.207
Log(theta)       0.358188   0.049009   7.309 2.70e-13 ***
Zero hurdle model coefficients (binomial with logit link):
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -0.09874    0.16296  -0.606 0.544584
hospital         0.36432    0.11052   3.296 0.000979 ***
healthexcellent -0.38506    0.16052  -2.399 0.016450 *
healthpoor       0.06522    0.18916   0.345 0.730252
chronic          0.53343    0.05195  10.268  < 2e-16 ***
sexmale         -0.38411    0.10111  -3.799 0.000145 ***
school           0.06835    0.01392   4.911 9.08e-07 ***
insuranceyes     0.75178    0.11723   6.413 1.43e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Theta: count = 1.4307
Number of iterations in BFGS optimization: 15
Log-likelihood: -9058 on 17 Df
```

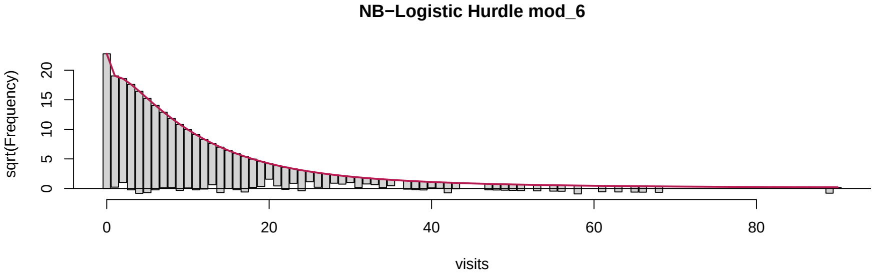
Rootogram for NB-Logistic Hurdle model

```
countreg::rootogram(mod_6, max = 90,
                    main = "NB-Logistic Hurdle: mod_6")
```



NB-Based Rootograms - Which Looks Best?





Store Training Sample `mod_6` Predictions

```r
mod_6_aug <- med_train %>%
    mutate(".fitted" = predict(mod_6, type = "response"),
           ".resid" = resid(mod_6, type = "response"))

mod_6_aug %>% select(subject, visits, .fitted, .resid) %>%
  head(3)
```

```
# A tibble: 3 x 4
  subject visits .fitted .resid
    <int>  <int>   <dbl>  <dbl>
1       1      5    5.96 -0.964
2       2      1    5.79 -4.79
3       3     13   18.3  -5.30
```

Summarizing Training Sample `mod_6` Fit

`mod_6_aug` has actual (`visits`) and `mod_6` predicted (in `.fitted`) counts.

```r
mod_6_summary <- tibble(
  model = "mod_6 (NB Hurdle)",
  R2 = R2(mod_6_aug$.fitted, mod_6_aug$visits),
  RMSE = RMSE(mod_6_aug$.fitted, mod_6_aug$visits),
  MAE = MAE(mod_6_aug$.fitted, mod_6_aug$visits))

mod_6_summary %>% kable(digits = 3)
```

model	R2	RMSE	MAE
mod_6 (NB Hurdle)	0.096	6.648	4.129

Training Sample Results through mod_6

```
bind_rows(mod_1_summary, mod_2_summary,
          mod_3_summary, mod_4_summary,
          mod_5_summary, mod_6_summary) %>%
  kable(digits = 3)
```

model	R2	RMSE	MAE
mod_1 (Poisson)	0.102	6.522	4.124
mod_2 (Neg. Binomial)	0.084	6.834	4.179
mod_3 (ZIP)	0.113	6.481	4.093
mod_4 (ZINB)	0.101	6.592	4.111
mod_5 (Poisson Hurdle)	0.113	6.481	4.093
mod_6 (NB Hurdle)	0.096	6.648	4.129

Vuong test: Comparing mod_4 and mod_6

```
vuong(mod_4, mod_6)
```

```
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A p-value
Raw                   0.1589962 model1 > model2 0.43684
AIC-corrected         0.1589962 model1 > model2 0.43684
BIC-corrected         0.1589962 model1 > model2 0.43684
```

There's some evidence mod_4 (ZINB) fits better than mod_6 (NB Hurdle) in our training sample, but not to a statistically significant degree, based on the large *p* value.

Cross-Validation

Validation: Test Sample Predictions

Predict the visit counts for each subject in our test sample.

- Use mod_1 as a model for mod_2.
- Use mod_3 as a model for mod_4, mod_5 and mod_6.
- The other models are included with echo = FALSE.

```r
test_1_aug <- augment(mod_1, newdata = med_test,
                      type.predict = "response")

test_2_aug <- augment(mod_2, newdata = med_test,
                      type.predict = "response")

test_3_aug <- med_test %>%
    mutate(".fitted" = predict(mod_3, newdata = med_test,
                        type = "response"))
```

Validation: Test Sample Fit Summaries

I'll show mod_1 and mod_2. The others are in the code with echo = FALSE.

```
mod_1_val <- tibble(
  model = "mod_1 (Poisson)",
  R2 = R2(test_1_aug$.fitted, test_1_aug$visits),
  RMSE = RMSE(test_1_aug$.fitted, test_1_aug$visits),
  MAE = MAE(test_1_aug$.fitted, test_1_aug$visits))

mod_2_val <- tibble(
  model = "mod_2 (Negative Binomial)",
  R2 = R2(test_2_aug$.fitted, test_2_aug$visits),
  RMSE = RMSE(test_2_aug$.fitted, test_2_aug$visits),
  MAE = MAE(test_2_aug$.fitted, test_2_aug$visits))
```

Results on the Next Slide

Validation Results in Test Sample: All Models

```
bind_rows(mod_1_val, mod_2_val, mod_3_val,
          mod_4_val, mod_5_val, mod_6_val) %>%
  kable(digits = 3)
```

model	R2	RMSE	MAE
mod_1 (Poisson)	0.085	6.156	4.159
mod_2 (Negative Binomial)	0.075	6.332	4.221
mod_3 (ZIP)	0.089	6.143	4.143
mod_4 (ZINB)	0.083	6.218	4.171
mod_5 (Poisson Hurdle)	0.089	6.143	4.143
mod_6 (NB Hurdle)	0.081	6.241	4.179

Now which model would you choose based on test sample performance?

Next Time

Modeling Multi-Categorical Outcomes