The Mathematica® Journal

Negative Binomial Regression

Michael L. Zwilling

Negative binomial regression is implemented using maximum likelihood estimation. The traditional model and the rate model with offset are demonstrated, along with regression diagnostics.

■ Traditional Model

Negative binomial regression is a type of generalized linear model in which the dependent variable Y is a count of the number of times an event occurs. A convenient parametrization of the negative binomial distribution is given by Hilbe [1]:

$$p(y) = P(Y = y) = \frac{\Gamma(y + 1/\alpha)}{\Gamma(y+1)\,\Gamma(1/\alpha)} \left(\frac{1}{1+\alpha\,\mu}\right)^{1/\alpha} \left(\frac{\alpha\,\mu}{1+\alpha\,\mu}\right)^{y}, \tag{1}$$

where $\mu > 0$ is the mean of Y and $\alpha > 0$ is the heterogeneity parameter. Hilbe [1] derives this parametrization as a Poisson-gamma mixture, or alternatively as the number of failures before the $(1/\alpha)^{\text{th}}$ success, though we will not require $1/\alpha$ to be an integer.

The traditional negative binomial regression model, designated the NB2 model in [1], is

$$\ln \mu = \beta_0 + \beta_1\, x_1 + \beta_2\, x_2 + \cdots + \beta_p\, x_p, \tag{2}$$

where the predictor variables $x_1, x_2, \ldots, x_p$ are given, and the population regression coefficients $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ are to be estimated.

Given a random sample of n subjects, we observe for subject i the dependent variable y_i and the predictor variables $x_{1i}, x_{2i}, \ldots, x_{pi}$. Utilizing vector and matrix notation, we let $\beta = (\,\beta_0 \;\; \beta_1 \;\; \beta_2 \;\; \cdots \;\; \beta_p\,)^{\mathrm{T}}$, and we gather the predictor data into the design matrix X as follows:

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}.$$

Designating the i^{th} row of X to be x_i, and exponentiating (2), we can then write the distribution (1) as

$$p(y_i) = \frac{\Gamma(y_i + 1/\alpha)}{\Gamma(y_i + 1)\,\Gamma(1/\alpha)} \left(\frac{1}{1 + \alpha\, e^{x_i \cdot \beta}}\right)^{1/\alpha} \left(\frac{\alpha\, e^{x_i \cdot \beta}}{1 + \alpha\, e^{x_i \cdot \beta}}\right)^{y_i},\ i = 1,\ 2,\ \ldots,\ n.$$

We estimate α and β using maximum likelihood estimation. The likelihood function is

$$L(\alpha, \beta) = \prod_{i=1}^{n} p(y_i) = \prod_{i=1}^{n} \frac{\Gamma(y_i + 1/\alpha)}{\Gamma(y_i + 1)\,\Gamma(1/\alpha)} \left(\frac{1}{1 + \alpha\, e^{x_i \cdot \beta}}\right)^{1/\alpha} \left(\frac{\alpha\, e^{x_i \cdot \beta}}{1 + \alpha\, e^{x_i \cdot \beta}}\right)^{y_i},$$

and the log-likelihood function is

$$\ln L(\alpha, \beta) = \sum_{i=1}^{n} \left(y_i \ln \alpha + y_i\,(x_i \cdot \beta) - \left(y_i + \frac{1}{\alpha}\right) \ln\left(1 + \alpha\, e^{x_i \cdot \beta}\right) + \ln \Gamma\left(y_i + \frac{1}{\alpha}\right) - \ln \Gamma(y_i + 1) - \ln \Gamma\left(\frac{1}{\alpha}\right)\right). \tag{3}$$

The values of α and β that maximize $\ln L(\alpha, \beta)$ will be the maximum likelihood estimates we seek, and the estimated variance-covariance matrix of the estimators is $\Sigma = -H^{-1}$, where H is the Hessian matrix of second derivatives of the log-likelihood function. Then the variance-covariance matrix can be used to find the usual Wald confidence intervals and p-values of the coefficient estimates.

Example 1: Traditional Model with Simulated Data

We will use *Mathematica* to replicate some examples given by Hilbe [1], who uses R and Stata. We start with simulated data generated with known regression coefficients, then recover the coefficients using maximum likelihood estimation. We will generate a sample of $n = 5000$ observations of a dependent random variable Y that has a negative binomial distribution with mean given by (2), using $p = 2$, $\alpha = 0.50$, and $\beta = (2.00\ .75\ -1.25)^{\mathrm{T}}$. The design matrix X_1 will contain independent standard normal variates.

```
SeedRandom[12 345];
n1 = 5000;
X1 = Prepend[#, 1] & /@
   RandomVariate[NormalDistribution[], {n1, 2}];
y1 =
  Table[RandomVariate[
    NegativeBinomialDistribution[1/α, 1/(1 + α ℯ^(X1[[i]].β))] /.
     {α → 0.50,
      β → {2.00, 0.75, -1.25}}], {i, n1}];
```

Now we define and maximize the log-likelihood function (3), obtaining the estimates of α and β. Some experimentation with starting values for the search may be required, and the accuracy goal may need to be lowered; we could obtain good starting values for β using Poisson regression via `GeneralizedLinearModelFit`, while α is usually between 0.0 and 4.0 [1].

```
GeneralizedLinearModelFit[{X1, y1},
 ExponentialFamily → "Poisson"]
```

```
FittedModel[ e^(2.01709 #1+0.726365 #2-1.25206 #3) ]
```

But we arbitrarily set all starting values to 1.0 and successfully find the correct estimates.

```
lnL1[X_, y_, α_, β_] :=
 Module[{n = Length[y]},
  Sum[y[[i]] Log[α] + y[[i]] (X[[i]].β) - (y[[i]] + 1/α) Log[1 + α e^(X[[i]].β)] +
      LogGamma[y[[i]] + 1/α] - LogGamma[y[[i]] + 1] - LogGamma[1/α]), {i, 1, n}]]

results1 = FindMaximum[lnL1[X1, y1, α, β],
  {{α, 1.0, 0.01, 4.0}, {β, {1.0, 1.0, 1.0}}},
  AccuracyGoal → 6]
```

```
{-15 429.2, {α → 0.49929, β → {2.00163, 0.755827, -1.25551}}}
```

Define two helper functions.

```
βs[X_] := Array[β, Length[X[[2]]], 0]

flattenrules[{_, {xx_ → xxval_, yy_ → yyvals_}}] :=
 {xx -> xxval,
   Thread[Array[yy, Length[yyvals], 0] -> yyvals]} // Flatten

flattenrules[results1]
```

```
{α → 0.49929, β[0] → 2.00163, β[1] → 0.755827, β[2] → -1.25551}
```

Next, we find the standard errors of the estimates. The standard errors are the square roots of the diagonal elements of the variance-covariance matrix Σ, which as mentioned above is given by $\Sigma = -H^{-1}$, where H is the Hessian matrix of second derivatives of the log-likelihood function. First, define the Hessian for any function.

```
HessianH[f_, x_] := D[f, {Flatten[x], 2}]
```

Then we find the Hessian and Σ at the values of our parameter estimates.

```
VarianceCovarianceMatrix[X_, y_, results_] :=
 -Inverse[
   HessianH[lnL1[X, y, α, β] /. β → βs[X], {α, βs[X]}] /.
    flattenrules[results]]

(v1 = VarianceCovarianceMatrix[X1, y1, results1]) //
 MatrixForm
```

$$\begin{pmatrix} 0.000173092 & -7.39458\times10^{-7} & 1.02221\times10^{-6} & -1.05205\times10^{-6} \\ -7.39458\times10^{-7} & 0.000155043 & -0.0000266927 & 0.0000479428 \\ 1.02221\times10^{-6} & -0.0000266927 & 0.00015131 & -0.0000161681 \\ -1.05205\times10^{-6} & 0.0000479428 & -0.0000161681 & 0.000168057 \end{pmatrix}$$

Finally, these are our standard errors.

```
StdErr1 = √Diagonal[v1]

{0.0131564, 0.0124516, 0.0123008, 0.0129637}
```

We can now print a table of the results: the estimates of the coefficients, their standard errors, and the Wald z statistics, p-values, and confidence intervals.

```
coefficients1 = Last /@ flattenrules[results1]

{0.49929, 2.00163, 0.755827, -1.25551}


CoefficientsTable[coefficients_, StdErr_, labels_] :=
 Module[{zStatistics, pValues, displaypValues, lower,
   upper},
  zStatistics = coefficients / StdErr;
  pValues = 2 CDF[NormalDistribution[], -Abs[zStatistics]];
  displaypValues =
   Table[{NumberForm[Round[pValues⟦j⟧, 0.0001], {5, 4}]},
     {j, Length[labels]}] // Flatten;
  lower = coefficients - 1.96 StdErr;
  upper = coefficients + 1.96 StdErr;
```

```
Text@
 Column[
  {TableForm[
    {coefficients, StdErr, zStatistics, displaypValues} //
     Transpose, TableHeadings →
     {labels, {"Estimate", "Std. Err.",
       Row[{Style["z", Italic], "-Statistic"}],
       TraditionalForm[P > Abs[z]]}}],
   "",
   TableForm[{coefficients, lower, upper} // Transpose,
    TableHeadings →
     {labels, {"Estimate", "95% CI lower",
       "95% CI upper"}}]
  }]
]
```

```
CoefficientsTable[coefficients1, StdErr1, Prepend[βs[X1], α]]
```

	Estimate	Std. Err.	z-Statistic	$P > \|z\|$
α	0.49929	0.0131564	37.9502	0.0000
$\beta[0]$	2.00163	0.0124516	160.753	0.0000
$\beta[1]$	0.755827	0.0123008	61.4453	0.0000
$\beta[2]$	−1.25551	0.0129637	−96.8481	0.0000

	Estimate	95% CI lower	95% CI upper
α	0.49929	0.473503	0.525077
$\beta[0]$	2.00163	1.97723	2.02604
$\beta[1]$	0.755827	0.731717	0.779936
$\beta[2]$	−1.25551	−1.28092	−1.2301

We see that in each case the confidence interval has captured the population parameter.

■ Traditional Model for Rates, Using Offset

If the dependent variable Y counts the number of events during a specified time interval t, then the observed rate Y/t can be modeled by using the traditional negative binomial model above, with a slight adjustment. We note that t can also be thought of as area or sub-population size, among other interpretations that lead to considering Y/t a rate.

Since $E(Y/t) = \mu/t$, we make the following adjustment to model (2) above:

$$\ln(\mu/t) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p,$$

which can also be written as:

$$\ln \mu = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \ln t. \tag{4}$$

This last term, $\ln t$, is called the *offset*. So in our log-likelihood function, instead of replacing μ with $e^{x_i \cdot \beta}$, we replace μ with $e^{x_i \cdot \beta + \ln t}$, resulting in the following:

$$\ln L(\alpha, \beta) = \sum_{i=1}^{n} \left(y_i \ln \alpha + y_i (x_i \cdot \beta + \ln t_i) - \left(y_i + \frac{1}{\alpha} \right) \ln\left(1 + \alpha\, e^{x_i \cdot \beta + \ln t_i}\right) + \ln \Gamma\left(y_i + \frac{1}{\alpha} \right) - \ln \Gamma(y_i + 1) - \ln \Gamma\left(\frac{1}{\alpha} \right) \right). \tag{5}$$

Then we proceed as before, maximizing the new log-likelihood function in order to estimate the parameters.

Example 2: Traditional Model with Offset for the *Titanic* Data

The *Titanic* survival data, available from [2] and analyzed in [1] using R and Stata, is summarized in Table 1, with crew members deleted.

Survived	*Cases*	*Age*	*Sex*	*Class*
14	31	child	female	third
13	13	child	female	second
1	1	child	female	first
13	48	child	male	third
11	11	child	male	second
5	5	child	male	first
76	165	adult	female	third
80	93	adult	female	second
140	144	adult	female	first
75	462	adult	male	third
14	168	adult	male	second
57	175	adult	male	first

▲ **Table 1.** *Titanic* survival dataset.

Why did fewer first-class children survive than second class or third class? Was it because first-class children were at extra risk? No, it was because there were fewer first-class children on board the *Titanic* in the first place. So we do not want to model the raw number (Y) of survivors; instead, we want to model the proportion (Y / cases) of survivors, which is the survival rate. So in (4) we need t to be the number of cases.

We set up the design matrix, with indicators 1 for adults and males, and using indicator variables for second class and third class, which means first class will be a reference.

```
ones = ConstantArray[1, 12];
age = {0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1};
sex = {0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1};
class2 = {0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0};
class3 = {1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0};
X2 = Join[Thread[List[ones, age, sex, class2, class3]]];
```

Then we set up the dependent variable and the offset.

```
y2 = {14, 13, 1, 13, 11, 5, 76, 80, 140, 75, 14, 57};
t2 = {31, 13, 1, 48, 11, 5, 165, 93, 144, 462, 168, 175};
```

We define the log-likelihood (5).

```
lnL2[X_, y_, t_, α_, β_] :=
 Module[{n = Length[y]},
  Sum[y[[i]] Log[α] + y[[i]] (X[[i]].β + Log[t[[i]]]) -
      (y[[i]] + 1/α) Log[1 + α ℯ^(X[[i]].β+Log[t[[i]]])] + LogGamma[y[[i]] + 1/α] -
      LogGamma[y[[i]] + 1] - LogGamma[1/α], {i, 1, n}]
 ]
```

Now we maximize it to find the coefficients.

```
results2 = FindMaximum[lnL2[X2, y2, t2, α, β],
  {{α, 1.0, 0.01, 4.0}, {β, {1.0, 1.0, 1.0, 1.0, 1.0}}},
  AccuracyGoal → 6]
```

```
{-43.7168, {α → 0.104034,
  β → {0.613375, -0.670035, -0.98015, -0.374614, -0.907064}}}
```

Then we find the standard errors of the coefficients.

```
VarianceCovarianceMatrix2[X_, y_, results_] :=
 -Inverse[
   HessianH[lnL2[X, y, t2, α, β] /. β → βs[X], {α, βs[X]}] /.
    flattenrules[results]]

StdErr2 =
 √Diagonal[VarianceCovarianceMatrix2[X2, y2, results2]]

{0.0683913, 0.32834, 0.2535, 0.245967, 0.30709, 0.287539}
```

And again we can print a table of the results.

```
CoefficientsTable[Last /@ flattenrules[results2],
 StdErr2, Prepend[βs[X2], α]]
```

	Estimate	Std. Err.	z-Statistic	$P > \|z\|$
α	0.104034	0.0683913	1.52116	0.1282
$\beta[0]$	0.613375	0.32834	1.86811	0.0617
$\beta[1]$	−0.670035	0.2535	−2.64313	0.0082
$\beta[2]$	−0.98015	0.245967	−3.98488	0.0001
$\beta[3]$	−0.374614	0.30709	−1.21988	0.2225
$\beta[4]$	−0.907064	0.287539	−3.15458	0.0016

	Estimate	95% CI lower	95% CI upper
α	0.104034	−0.0300126	0.238081
$\beta[0]$	0.613375	−0.0301722	1.25692
$\beta[1]$	−0.670035	−1.1669	−0.173175
$\beta[2]$	−0.98015	−1.46225	−0.498054
$\beta[3]$	−0.374614	−0.97651	0.227283
$\beta[4]$	−0.907064	−1.47064	−0.343488

But perhaps more useful for interpretation of the coefficients would be the Incidence Rate Ratio (IRR) for each variable, which is obtained by exponentiating each coefficient. For example, out of a sample of t adults, we expect that the survival rate, from our model (4), will be $\mu_{\text{adults}}/t = \exp(\beta_0 + \beta_1(1) + \beta_2(\text{sex}) + \beta_2(\text{class2}) + \beta_3(\text{class3}))$, while for an identical number t of children we expect their survival rate to be $\mu_{\text{children}}/t = \exp(\beta_0 + \beta_1(0) + \beta_2(\text{sex}) + \beta_2(\text{class2}) + \beta_3(\text{class3}))$. So by dividing the two rates, we obtain the ratio of rates (IRR) to be

$$\text{IRR} = \frac{\mu_{\text{adults}}/t}{\mu_{\text{children}}/t} = \frac{\exp(\beta_0 + \beta_1(1) + \beta_2(\text{sex}) + \beta_2(\text{class2}) + \beta_3(\text{class3}))}{\exp(\beta_0 + \beta_1(0) + \beta_2(\text{sex}) + \beta_2(\text{class2}) + \beta_3(\text{class3}))} = e^{\beta_1},$$

which we estimate to be $e^{-0.670034} = 0.51$. Thus, our interpretation is that adults survived at roughly half the rate at which children survived, among those of the same sex and class.

The standard error of IRR is found by multiplying the estimated IRR by the standard error of the coefficient (see [1]), while a confidence interval for IRR is found by exponentiating the confidence interval for the coefficient. Thus we obtain the following.

```
coefficients2 = Last /@ flattenrules[results2];
lower2 = coefficients2 - 1.96 StdErr2;
upper2 = coefficients2 + 1.96 StdErr2;

IRR = e^coefficients2;
IRRStdErr = IRR StdErr2;
IRRlower = e^lower2;
IRRupper = e^upper2;
```

We do not need IRR for α or β_0, so we drop them and then print the resulting table.

```
IRR = Drop[IRR, 2];
IRRStdErr = Drop[IRRStdErr, 2];
IRRlower = Drop[IRRlower, 2];
IRRupper = Drop[IRRupper, 2];
Text@
 TableForm[{IRR, IRRStdErr, IRRlower, IRRupper} // Transpose,
  TableHeadings → {{"age", "sex", "class2", "class3"},
    {"IRR", "IRR\nStd. Err.", "95% IRR\nCI lower",
     "95% IRR\nCI upper"}}]
```

	IRR	IRR Std. Err.	95% IRR CI lower	95% IRR CI upper
age	0.511691	0.129714	0.311332	0.840991
sex	0.375255	0.0923003	0.231715	0.607712
class2	0.687555	0.211141	0.376623	1.25518
class3	0.403708	0.116082	0.229778	0.709292

The confidence interval for the variable `class2` contains 1.0, consistent with the lack of significance of its coefficient, and indicating that the survival rate of second-class passengers was not significantly different than that of first-class passengers. We will address this after computing some model assessment statistics and residuals.

Model Assessment

Various types of model fit statistics and residuals are readily computed. We use definitions given in [1]; alternate definitions exist and would require only minor changes.

Commonly used model fit statistics include the log-likelihood, deviance, Pearson chi-square dispersion, Akaike Information Criterion (AIC), and Bayesian Information Criterion (BIC).

We already have the log-likelihood $\mathcal{L}$ as a byproduct of the maximization process. The deviance D is defined as

$$D = 2\sum_{i=1}^{n}(\mathcal{L}(y_i; y_i) - \mathcal{L}(\mu_i; y_i)),$$

where $\mathcal{L}(\mu_i; y_i)$ is our log-likelihood function (5), and $\mathcal{L}(y_i; y_i)$ is the log-likelihood function with y_i replacing μ_i. For our NB2 model, this simplifies to $D = \sum_{i=1}^{n} {d_i}^2$, where

$${d_i}^2 = \begin{cases} 2\left(y_i \ln\left(\frac{y_i}{\mu_i}\right) - \left(y_i + \frac{1}{\alpha}\right)\ln\left(\frac{1+\alpha\, y_i}{1+\alpha\,\mu_i}\right)\right) & \text{if } y_i > 0 \\ \frac{2}{\alpha}\ln(1+\alpha\,\mu_i) & \text{if } y_i = 0. \end{cases} \tag{6}$$

The Pearson chi-square dispersion statistic is given by $\sum_{i=1}^{n}(y_i - \mu_i)^2 \Big/ \left(\mu_i + \alpha\,{\mu_i}^2\right)$, while AIC and BIC are defined as

$$\text{AIC} = \frac{2\,(\mathcal{L} - (p+1))}{n}$$

and

$$\text{BIC} = D - (n - p - 1)\ln n.$$

We compute these for the *Titanic* data above and display them.

```
dSquared[i_, y_, μ_, A_] :=
 {2 (y[[i]] Log[y[[i]]/μ[[i]]] - (y[[i]] + 1/A) Log[(1+A y[[i]])/(1+A μ[[i]])])   y[[i]] ≠ 0
                  (2/A) Log[1 + A μ[[i]]]                                        y[[i]] == 0 ;
```

```
ModelAssessment[coefficients_, X_, y_, t_] :=
 Module[{μ, n, results, deviance, Pearson, p, AIC, BIC},
  μ = e^(X.Drop[coefficients,1]+Log[t]);
  n = Length@y;
  results = FindMaximum[lnL2[X, y, t, α, β],
    {{α, 1.0, 0.01, 4.0}, {β, Table[1.0, {Length@X[[1]]}]} },
    AccuracyGoal → 6];
  ℒ = results[[1]];
  deviance = Sum[dSquared[i, y, μ, coefficients[[1]]], {i, 1, n}];
  Pearson = Sum[(y[[i]] - μ[[i]])^2/(μ[[i]] + coefficients[[1]] μ[[i]]^2), {i, 1, n}];
  p = Length[X[[2]]] - 1;
  AIC = -(2 (ℒ - (p + 1))/n);
  BIC = deviance - (n - p - 1) Log[n];
  {ℒ, deviance, Pearson, AIC, BIC}
 ]

ModelAssessmentTable[coefficients_, X_, y_, t_] :=
 Text@TableForm[ModelAssessment[coefficients, X, y, t],
   TableHeadings →
    {{"Log-likelihood", "Deviance", "Pearson", "AIC",
      "BIC"}}]

ModelAssessmentTable[coefficients2, X2, y2, t2]
```

Log-likelihood	−43.7168
Deviance	12.4795
Pearson	11.0715
AIC	8.11947
BIC	−4.91485

These model assessment statistics are most useful when compared to those of a competing model, which we pursue in the next section after computing residuals.

Residuals

The raw residuals are of course $y_i - \mu_i$, while the Pearson residuals are $(y_i - \mu_i) \Big/ \sqrt{\mu_i + \alpha\, \mu_i^2}$, and the deviance residuals are $(\text{sgn}(y_i - \mu_i))\, d_i$, as defined in (6).

These residuals can be standardized by dividing by $\sqrt{1 - h_i}$, where the h_i are the leverages obtained from the diagonal of the hat matrix $W^{1/2} X (X' W X)^{-1} X' W^{1/2}$, for W equal to the $n \times n$ diagonal matrix, with $\mu_i / (1 + \alpha\, \mu_i)$ as the i^{th} element of the diagonal.

Here are the unstandardized residuals for the *Titanic* data.

```
UnstandardizedResiduals[coefficients_, X_, y_, t_] :=
 Module[{rawResiduals, μ, PearsonResiduals,
   DevianceResiduals},
  μ = e^(X.Drop[coefficients,1]+Log[t]);
  rawResiduals = y - μ;
  PearsonResiduals = (y - μ) / Sqrt[μ + coefficients[[1]] μ^2];
  DevianceResiduals =
   Table[Sign[y2[[i]] - μ[[i]]]
     Sqrt[dSquared[i, y, μ, coefficients[[1]]]] , {i, Length@y}];
  {rawResiduals, PearsonResiduals, DevianceResiduals}
 ]

UnstandardizedResidualsTable[coefficients_, X_, y_, t_] :=
 Text@
  TableForm[UnstandardizedResiduals[coefficients, X, y, t] //
    Transpose, TableHeadings →
    {{}, {"Raw\nresiduals", "Pearson\nresiduals",
      "Deviance \nresiduals"}}]
```

```
UnstandardizedResidualsTable[coefficients2, X2, y2,
 t2]
```

	Raw residuals	Pearson residuals	Deviance residuals
	−9.11076	−1.02715	−1.17269
	−3.50578	−0.523491	−0.557314
	−0.846654	−0.570629	−0.63435
	−0.428274	−0.075488	−0.0761329
	5.75903	2.0237	1.67408
	1.53517	0.707087	0.651037
	13.0575	0.599055	0.564238
	19.5796	0.9332	0.853039
	3.93218	0.0865899	0.0857962
	8.86544	0.388343	0.373189
	−26.9577	−1.83646	−2.4307
	−5.05222	−0.23489	−0.241116

And here are the leverages and the standardized residuals.

```
LeveragesAndStandardizedResiduals[coefficients_, X_,
  y_, t_] :=
 Module[{μ, h, PearsonResiduals, DevianceResiduals,
   StandardizedPearsonResiduals,
   StandardizedDevianceResiduals},
  μ = e^(X.Drop[coefficients,1]+Log[t]);
  W = DiagonalMatrix[μ/(1 + coefficients[[1]] μ)];
  (*W=DiagonalMatrix[μ/(1+α μ)]/.α->coefficients[[1]];*)
  h = Diagonal[W^(1/2).X.Inverse[Transpose[X].W.X].
     Transpose[X].W^(1/2)] ;
  {PearsonResiduals, DevianceResiduals} =
   Rest@UnstandardizedResiduals[coefficients, X, y, t];
  StandardizedPearsonResiduals = PearsonResiduals/Sqrt[1 - h];
  StandardizedDevianceResiduals = DevianceResiduals/Sqrt[1 - h];
  {h, StandardizedPearsonResiduals,
   StandardizedDevianceResiduals}
 ]
```

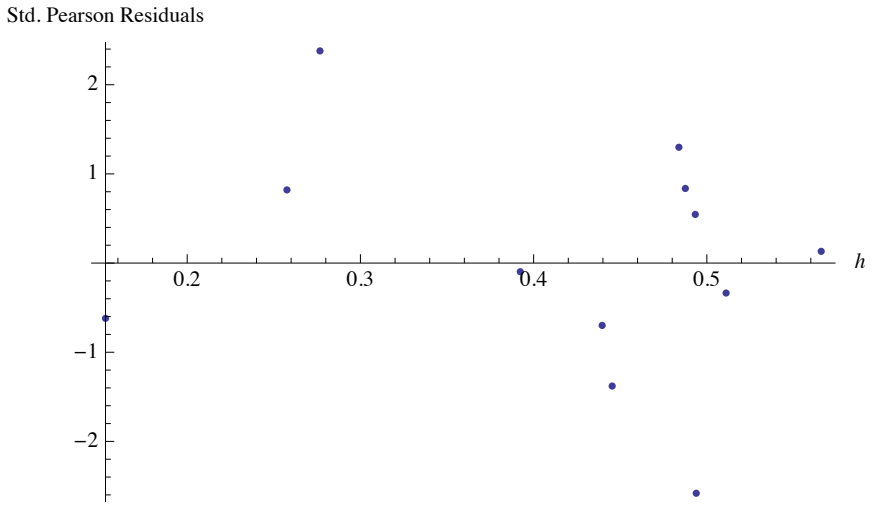
```
LeveragesAndStandardizedResidualsTable[coefficients_,
  X_, y_, t_] :=
 Text@
  TableForm[
   LeveragesAndStandardizedResiduals[coefficients,
     X, y, t] // Transpose,
   TableHeadings →
    {{}, {Row[{"Leverages\n(", Style["h", Italic], ")"}],
      "Standardized\nPearson\nresiduals",
      "Standardized\nDeviance\nresiduals"}}]

LeveragesAndStandardizedResidualsTable[coefficients2,
 X2, y2, t2]
```

	Leverages (h)	Standardized Pearson residuals	Standardized Deviance residuals
	0.445344	−1.37918	−1.57461
	0.439559	−0.699268	−0.744449
	0.152831	−0.619967	−0.689198
	0.392289	−0.0968343	−0.0976616
	0.276644	2.37941	1.96835
	0.257581	0.820632	0.755581
	0.487584	0.836864	0.788226
	0.483857	1.29894	1.18736
	0.566014	0.13144	0.130236
	0.493325	0.54557	0.52428
	0.493871	−2.58137	−3.41664
	0.511101	−0.335934	−0.344839

Hilbe recommends plotting the Standardized Pearson residuals versus h, with a poor model fit indicated by residuals that are outside the interval ± 2 when the leverage is high.

```
ListPlot[
 Most@LeveragesAndStandardizedResiduals[coefficients2,
    X2, y2, t2] // Transpose,
 AxesLabel → {Style["h", Italic], "Std. Pearson Residuals"}]
```



We have two Standardized Pearson residuals that are not within the range ± 2, one of which has a high leverage. We also recall that the variable `class2` was not significant. Perhaps the model will be improved if we remove `class2`. All that is required is to remove `class2` from the design matrix X, remove the corresponding starting value from the maximizing command, and run the model again. We obtain the following assessment statistics and standardized residuals for the revised model with `class2` removed.

We set up design matrix X and find the coefficients.

```
X3 = Join[Thread[List[ones, age, sex, class3]]];
results3 = FindMaximum[lnL2[X3, y2, t2, α, β],
  {{α, 1.0, 0.01, 4.0}, {β, {1.0, 1.0, 1.0, 1.0}}},
  AccuracyGoal → 6];
coefficients3 = Last /@ flattenrules[results3]
```

```
{0.133933, 0.365058, -0.614273, -0.917076, -0.729812}
```

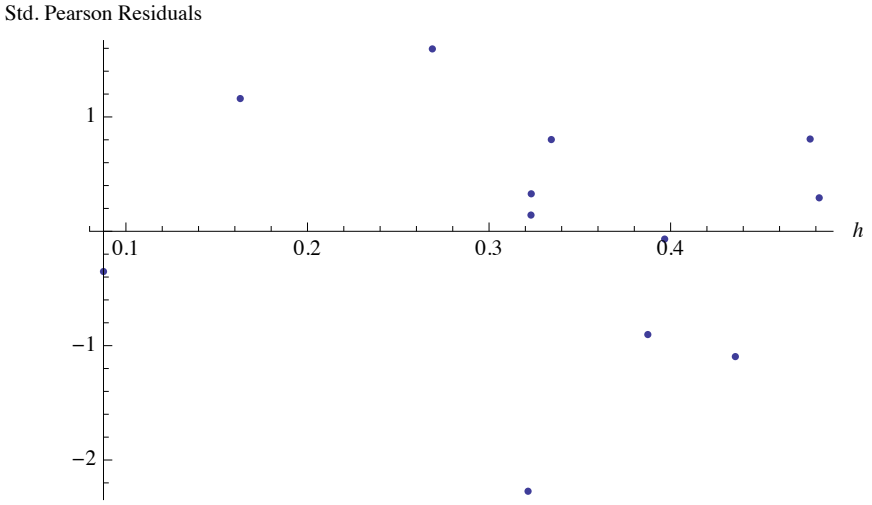
```
ModelAssessmentTable[coefficients3, X3, y2, t2]
```

Log-likelihood	−44.3705
Deviance	11.7129
Pearson	8.68622
AIC	8.06175
BIC	−8.1664

```
LeveragesAndStandardizedResidualsTable[coefficients3,
 X3, y2, t2]
```

	Leverages (h)	Standardized Pearson residuals	Standardized Deviance residuals
	0.435655	−1.09572	−1.2326
	0.387444	−0.902867	−0.996893
	0.0876473	−0.351868	−0.3759
	0.396719	−0.0679446	−0.0683983
	0.268772	1.59492	1.37705
	0.162966	1.16078	1.0196
	0.476846	0.806843	0.755576
	0.323239	0.327872	0.317654
	0.334295	0.802214	0.745825
	0.481837	0.292695	0.285484
	0.321482	−2.27311	−3.22693
	0.323096	0.141915	0.139939

```
ListPlot[
 Most@LeveragesAndStandardizedResiduals[coefficients3,
    X3, y2, t2] // Transpose,
 AxesLabel → {Style["h", Italic], "Std. Pearson Residuals"}]
```



Comparing to the full model, we see that the assessment statistics have improved (they are smaller, indicating a better fit), and the Standardized Pearson residuals with high leverages are within the recommended boundaries. It appears that the model has been improved by dropping `class2`.

Conclusion

The traditional negative binomial regression model (NB2) was implemented by maximum likelihood estimation without much difficulty, thanks to the maximization command and especially to the automatic computation of the standard errors via the Hessian.

Other negative binomial models, such as the zero-truncated, zero-inflated, hurdle, and censored models, could likewise be implemented by merely changing the likelihood function.

Acknowledgments

The author acknowledges suggestions and assistance by the editor and the referee that helped to improve this article.

References

[1] J. Hilbe, *Negative Binomial Regression*, 2nd ed., New York: Cambridge University Press, 2011.

[2] "JSE Data Archive." *Journal of Statistics Education*. (Nov 19, 2012) www.amstat.org/publications/jse/jse_data_archive.htm.

M. L. Zwilling, "Negative Binomial Regression," *The Mathematica Journal*, 2013. dx.doi.org/10.3888/tmj.15-6.

About the Author

Michael L. Zwilling
Department of Mathematics
University of Mount Union
1972 Clark Avenue
Alliance, OH 44601
zwilliml@mountunion.edu