Lecture 16

Peter Shaffery

3/3/2021

Logistic Regression Residuals

In linear regression, our choice of residual was fairly obvious:

$$\hat{\epsilon}_i = y_i - \hat{y}_i$$

However in logistic regression, as with R^2, we have a couple of options available to us for residuals.

One such option are the Pearson residuals (or chi-square residual):

$$\begin{aligned}\chi_i &= \frac{y_i - \hat{y}_i}{\sqrt{\text{Var}[y_i|\hat{\pi}_i]}} \\ &= \frac{y_i - n_i\hat{\pi}_i}{\sqrt{n_i\hat{\pi}_i(1-\hat{\pi}_i)}}\end{aligned}$$

This choice of residual follows from the Pearson chi-square goodness of fit measure:

$$\chi^2 = \sum_i \frac{(y_i - n_i\hat{\pi}_i)^2}{n_i\hat{\pi}_i(1-\hat{\pi}_i)}$$

Since it's not hard to see that:

$$\chi^2 = \sum_i \chi_i^2$$

For a detailed discussion of χ^2, see IGLM 7.5. We won't talk too much more about it, since for large n it's equivalent to the model deviance.

Another choice for residuals are *Deviance residuals*, which measure an individual observation's contribution to the overall model deviance.

For a normal model, we saw that the Deviance was:

$$D_{\text{Normal}} \propto \sum_i \hat{\epsilon}_i^2$$

This will motivate us to define our deviance residuals D_i such that $D = \sum_i D_i^2$.

For the Binomial regression model we showed the model deviance to be:

$$D = 2\sum_i \left[y_i \log \frac{y_i}{n_i\hat{\pi}_i} + (n_i - y_i)\log\frac{n_i - y_i}{n_i(1-\hat{\pi}_i)}\right]$$

Hence for a single observation y_i (along with n_i), the Deviance residual should be:

$$D_i = \sqrt{2\left[y_i \log \frac{y_i}{n_i\hat{\pi}_i} + (n_i - y_i)\log \frac{n_i - y_i}{n_i(1-\hat{\pi}_i)}\right]}$$

Note that this definition of D_i cannot be less than 0, which is a little weird. To fix this we amend the definition to be:

$$D_i = \text{sign}(y_i - n_i\hat{\pi}_i)\sqrt{2\left[y_i \log \frac{y_i}{n_i\hat{\pi}_i} + (n_i - y_i)\log \frac{n_i - y_i}{n_i(1-\hat{\pi}_i)}\right]}$$

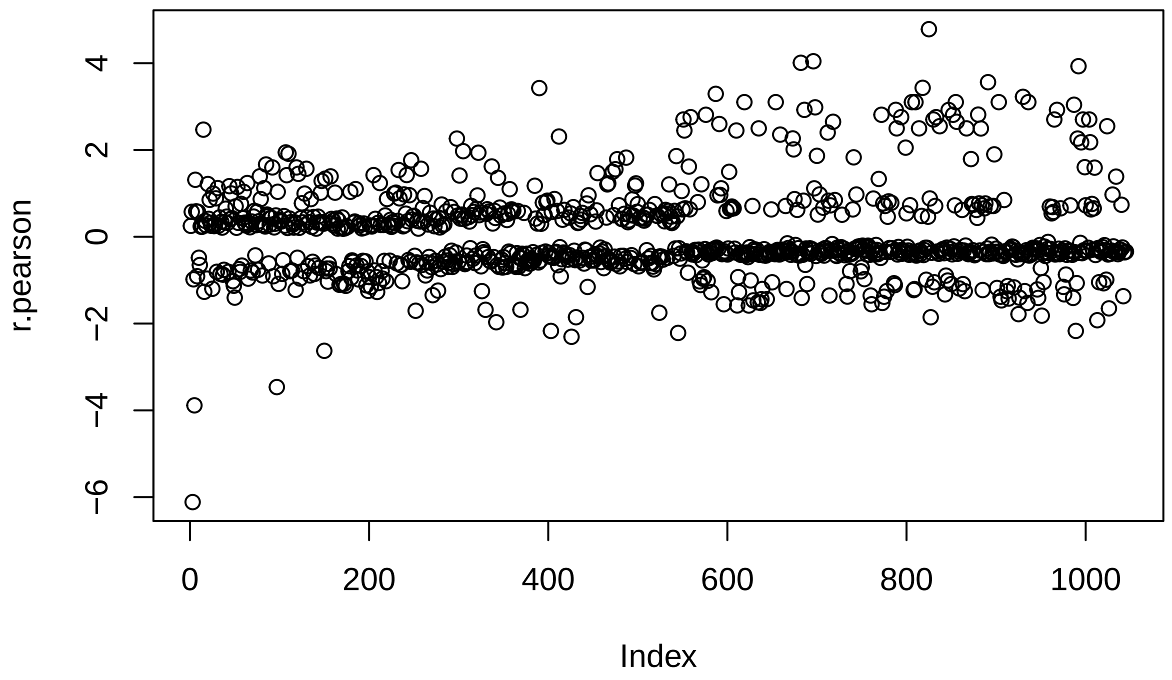
Observe that we still have $D = \sum_i D_i^2$.

Both the deviance and Pearson residuals, can be *standardized* using the leverage h_i (the i^{th} diagonal element of the hat matrix $H = X(X^TX)^{-1}X^T$):

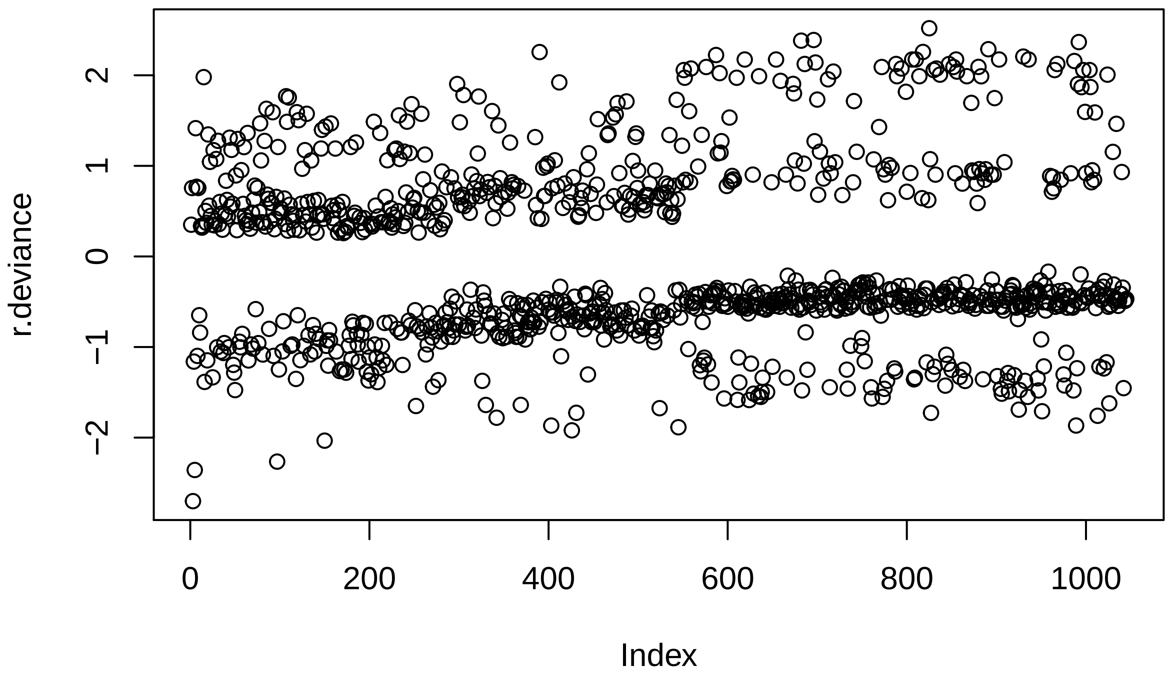
$$r_i^D = \frac{D_i}{\sqrt{1-h_i}}, r_i^\chi = \frac{\chi_i}{\sqrt{1-h_i}}$$

When the n_i are large, both types of standardized residuals will be approximately $N(0,1)$.

```r
library(tidyverse)
library(magrittr)

dat = read.csv('../../data/titanic.csv')

# the titanic data contains some missing data that we'll just ignore for now....
dat %<>% drop_na

# we don't care about a few of the columns
dat %<>% select(-c('name','ticket','cabin'))
dat$pclass %<>% as.factor

mod = glm(survived~pclass+age+sex+sibsp+parch, family=binomial, data=dat)

r.pearson = resid(mod,type='pearson')
plot(r.pearson)
```



```
r.deviance = resid(mod, type='deviance')
plot(r.deviance)
```



```
# Null model deviance is model containing only intercept, residual deviance is sum of squared residuals
summary(mod)
```

```
##
## Call:
## glm(formula = survived ~ pclass + age + sex + sibsp + parch,
##     family = binomial, data = dat)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -2.7009  -0.6644  -0.4202   0.6663   2.5192
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.904220   0.362085  10.783  < 2e-16 ***
## pclass2     -1.366033   0.230039  -5.938 2.88e-09 ***
## pclass3     -2.350481   0.228899 -10.269  < 2e-16 ***
## age         -0.039436   0.006639  -5.940 2.85e-09 ***
## sexmale     -2.556308   0.173269 -14.753  < 2e-16 ***
## sibsp       -0.352838   0.105340  -3.350  0.00081 ***
## parch        0.074320   0.099898   0.744  0.45690
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1413.57  on 1044  degrees of freedom
## Residual deviance:  970.05  on 1038  degrees of freedom
## AIC: 984.05
##
## Number of Fisher Scoring iterations: 4
```

```
sum(r.deviance^2)
```

```
## [1] 970.0529
```

Diagnostic Plots

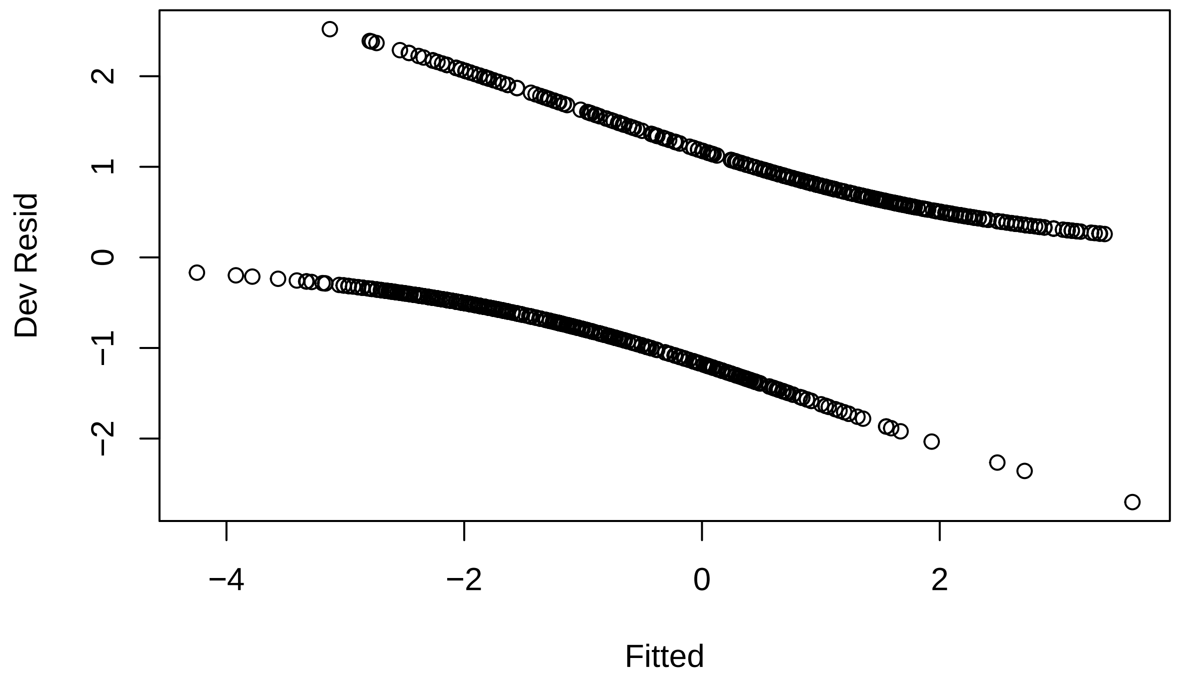
In general, diagnosing a GLM is more challenging than diagnosing a linear regression, it is typically recommended to place a lot of weight on things like deviance to get a handle of model performance. Nevertheless, we do still have some familiar tools at our disposal.

Residuals vs Fitted Values

As with linear regression, we want to determine whether our "linear model" is appropriate, that is whether $g(E[y_i])$ is indeed linear in the independent variables $\vec{x}_i$.

One way that we can do this is by plotting the (standardized) residuals against the "predicted value" of $g(E[y_i])$, $\vec{x}_i\hat{\beta}$:

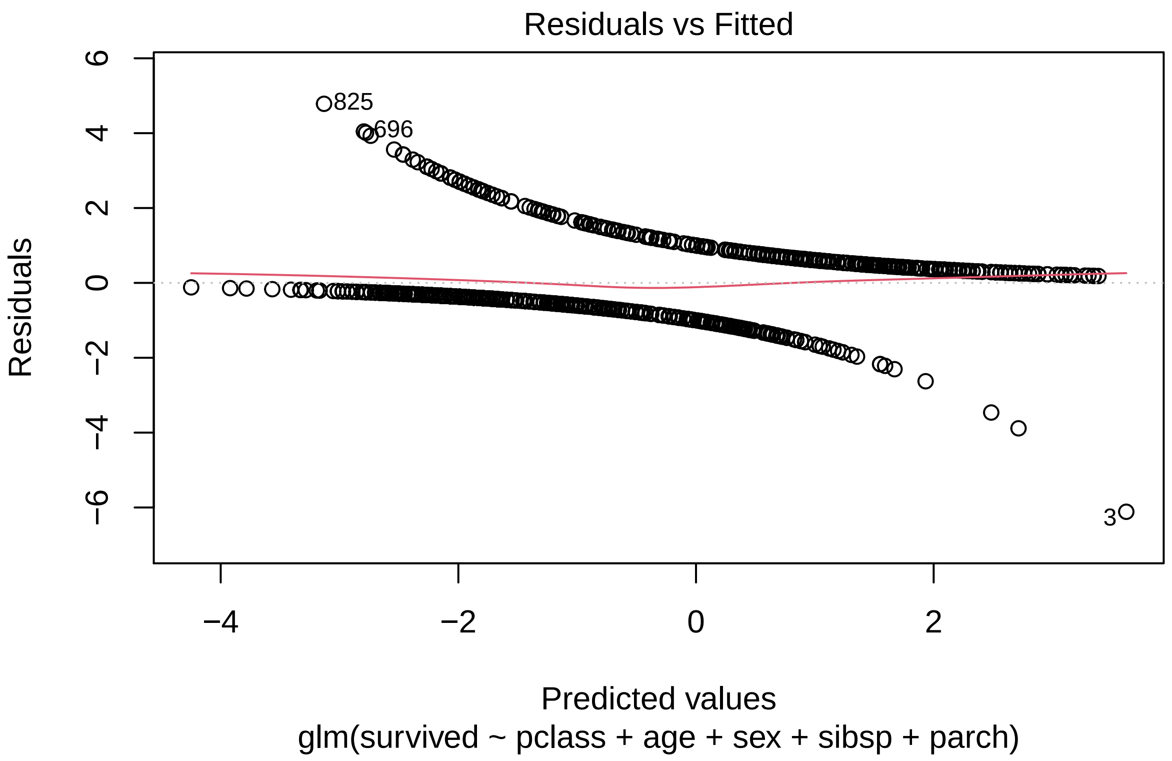
```
h = hatvalues(mod)
r.dev.std = r.deviance/sqrt(1-h)
plot(predict(mod),r.deviance, xlab='Fitted', ylab='Dev Resid')
```



Now, if this were a linear regression we would be very skeptical of this plot. However, recall that the deviance residuals are calculated from the y_i, which in this case can only be 0 or 1. This produces the "banding" that we see here.

In general, for a GLM we will always expect some trend due to the link function. What matters here overall is that there is not **average** trend, that is the residuals are not increasing or decreasing overall:

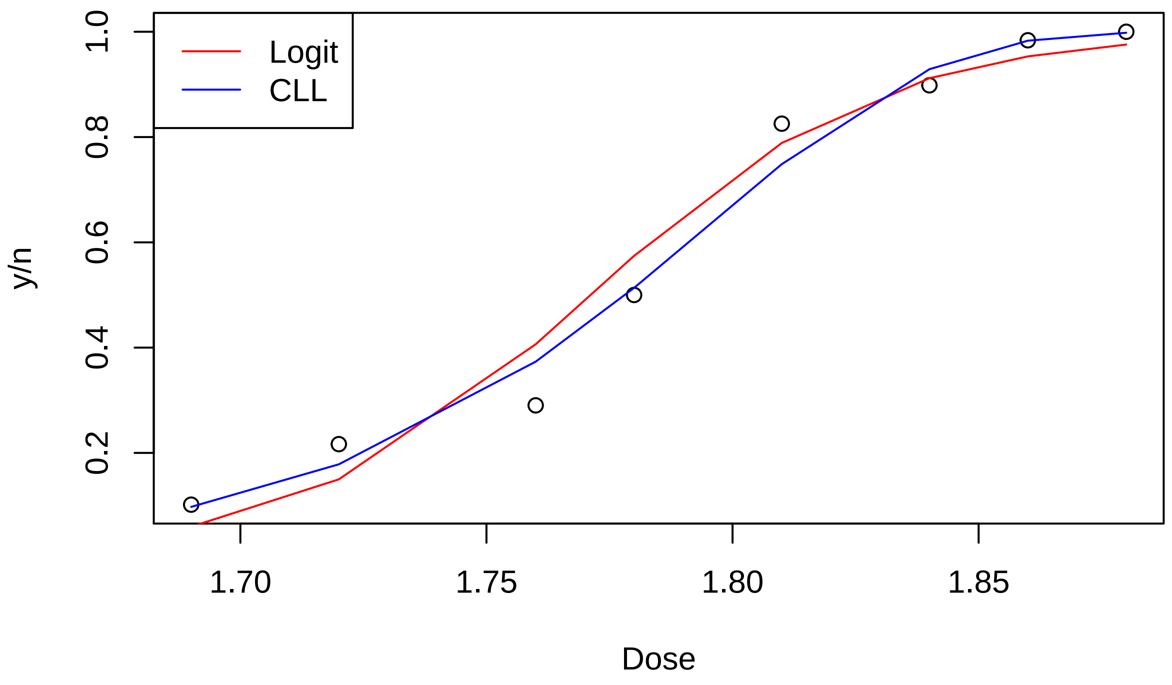
```
plot(mod,which=1)
```



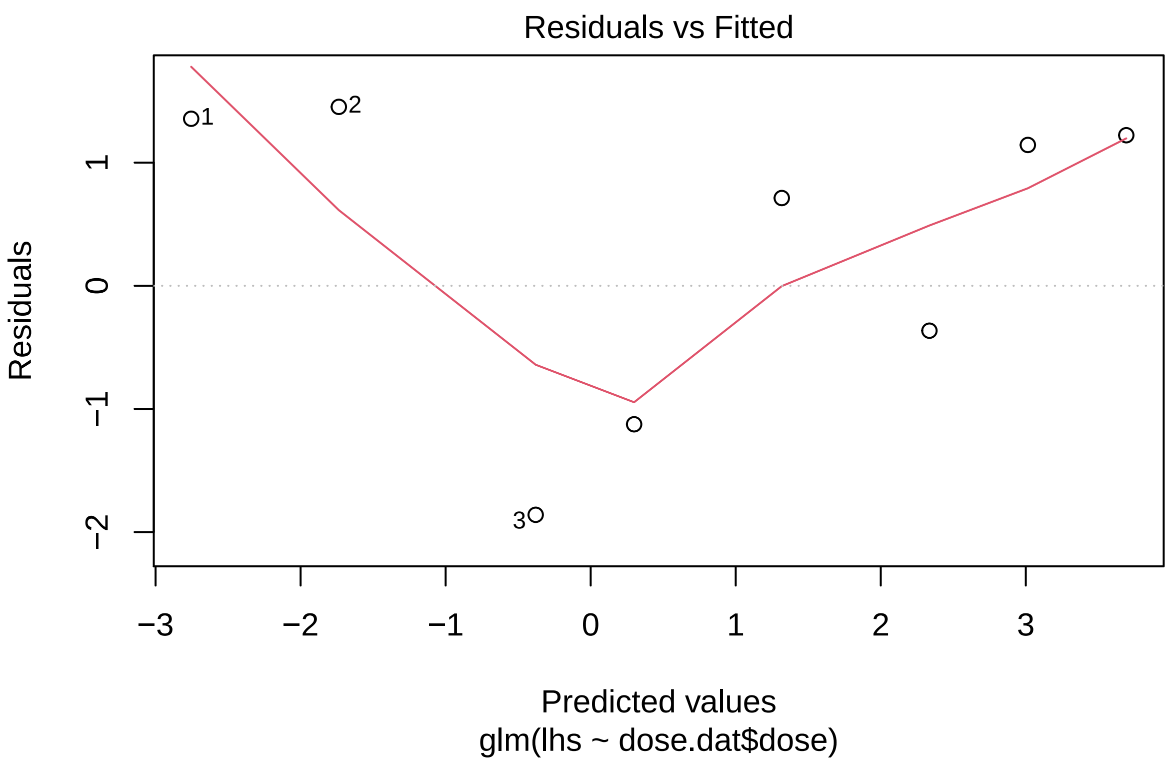
Be aware that R will give you the Pearson residuals. Almost always it makes no difference.

When average trend *does* exist, it indicates that a better choice of link function is required. Let's see dose-response example from Lecture 14:

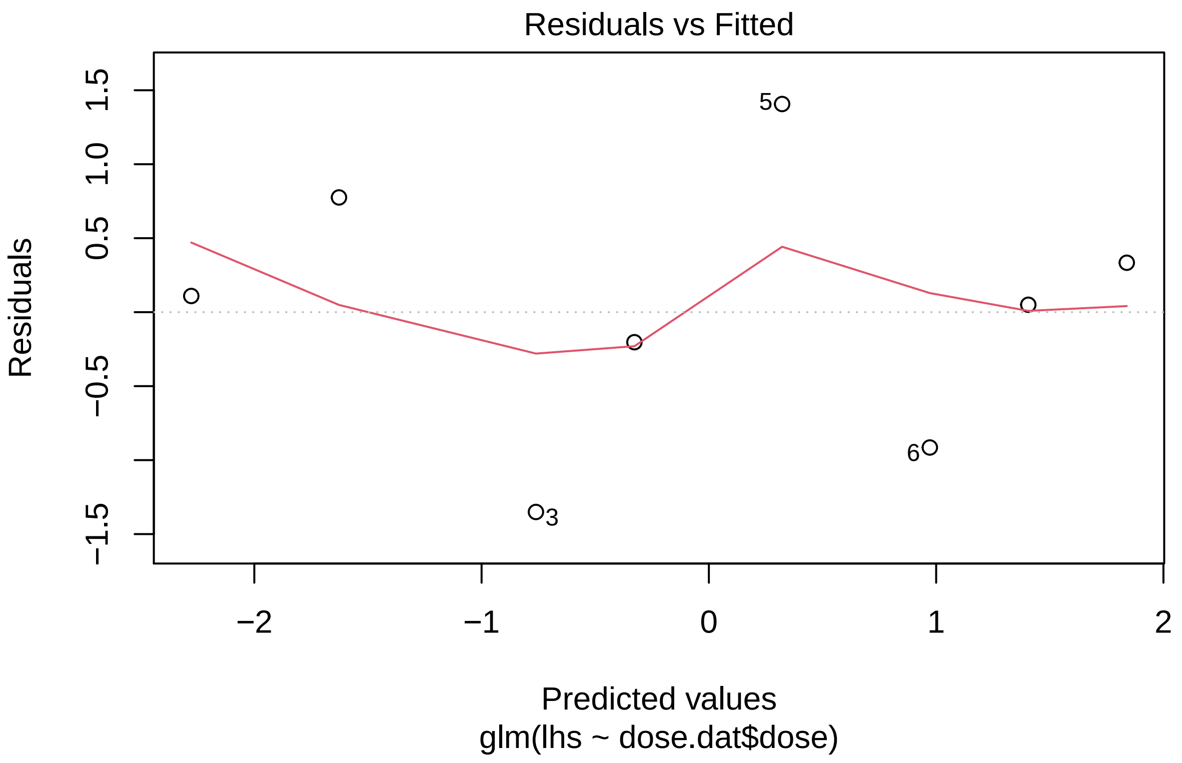
```
dose.dat = read.csv('../../data/beetles.csv')
dose.dat$fails = dose.dat$n - dose.dat$y
lhs = as.matrix(dose.dat[,c('y','fails')])

mod.logit = glm(lhs~dose.dat$dose, family=binomial)
mod.cll = glm(lhs~dose.dat$dose, family=binomial(link='cloglog'))

plot(dose.dat$dose,dose.dat$y/dose.dat$n,xlab='Dose',ylab='y/n')
lines(dose.dat$dose,predict(mod.logit, type='response'),col='red')
lines(dose.dat$dose,predict(mod.cll, type='response'),col='blue')
legend('topleft',c('Logit','CLL'),col=c('red','blue'), lwd=1)
```



```
plot(mod.logit,which=1)
```



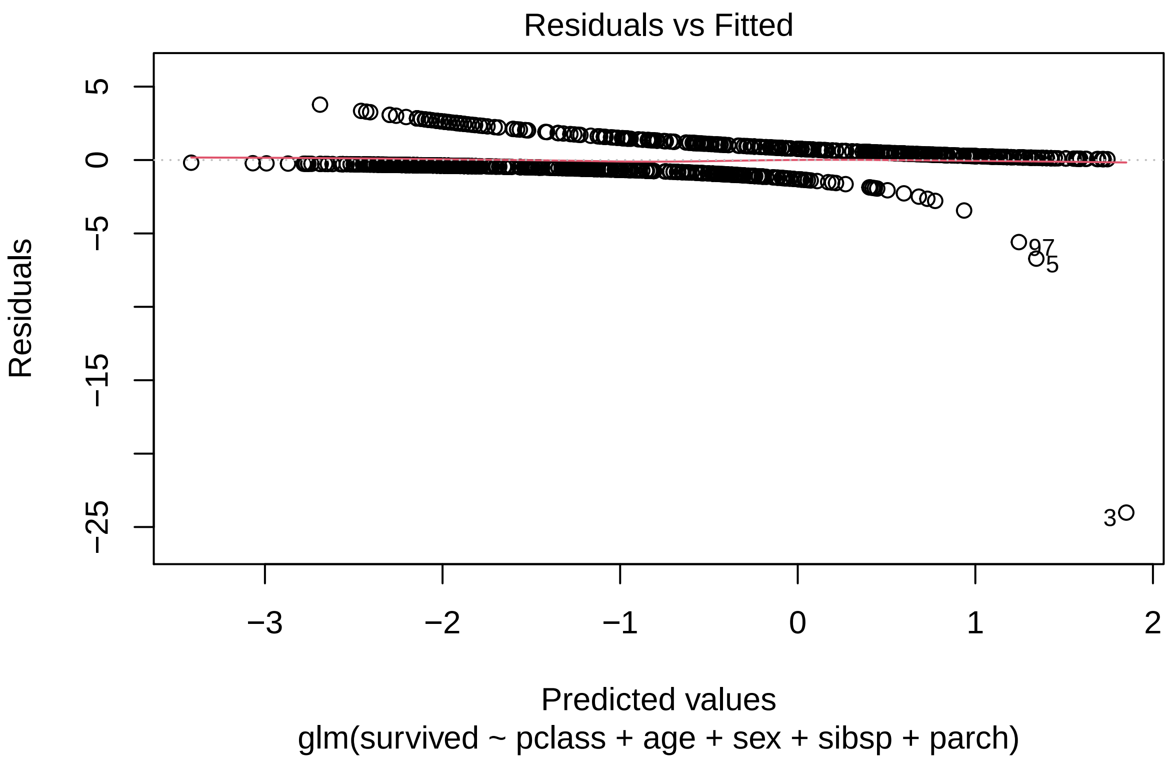
```r
plot(mod.cll,which=1)
```



Here we see that the complimentary log-log link (which produced the best fitting model last time we saw it) also produces a better residual-fitted plot than a logistic regression.

(PS: you might notice a similar, although slight, trend occurring in the Titanic plot as well, which might further lead you to conclude that a CLL model is better here as well…)

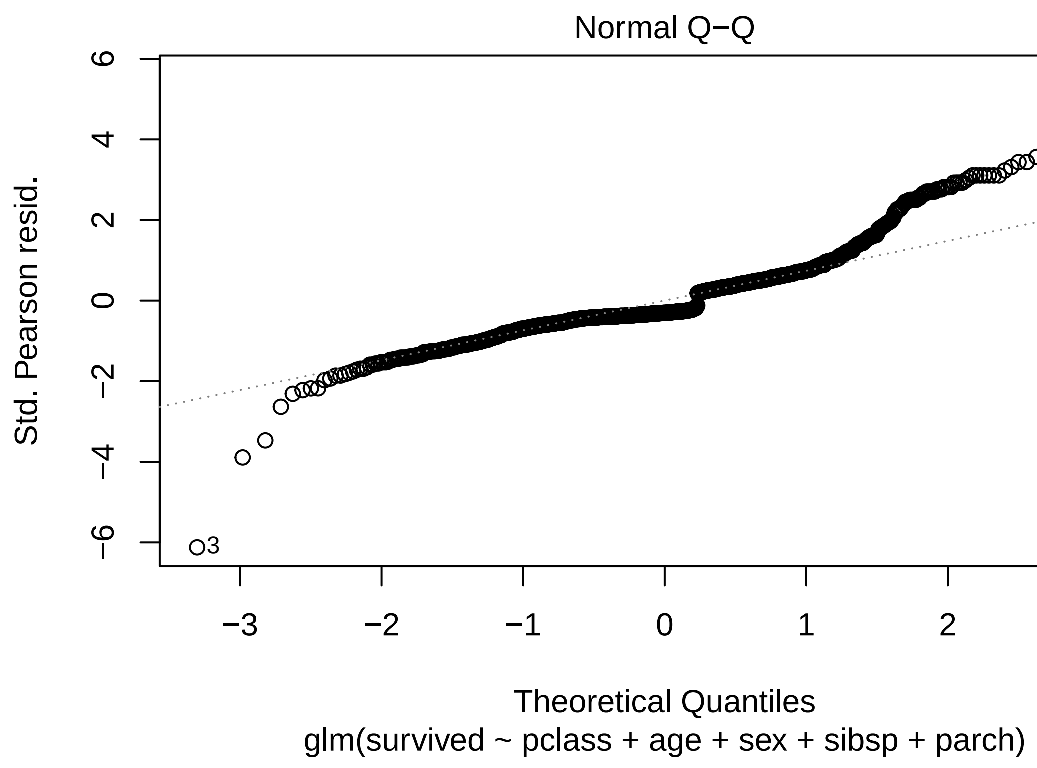
```
mod.cll = glm(survived~pclass+age+sex+sibsp+parch, family=binomial(link='cloglog'), data=dat)
plot(mod.cll,which=1)
```



Q-Q Plots

As mentioned above, standardized Pearson residuals are approximately normal:

```
plot(mod,which=2)
```



The problem here is that, while a good Q-Q plot is a good sign, a bad Q-Q plot can't really be interpreted. Either your choice of distribution was bad, or you have not achieved approximate normality yet. Just looking at the residuals it can be practically impossible to tell, and even harder to fix. For this reason, Q-Q plots aren't particularly informative for GLMs.

Scale-Location Plots

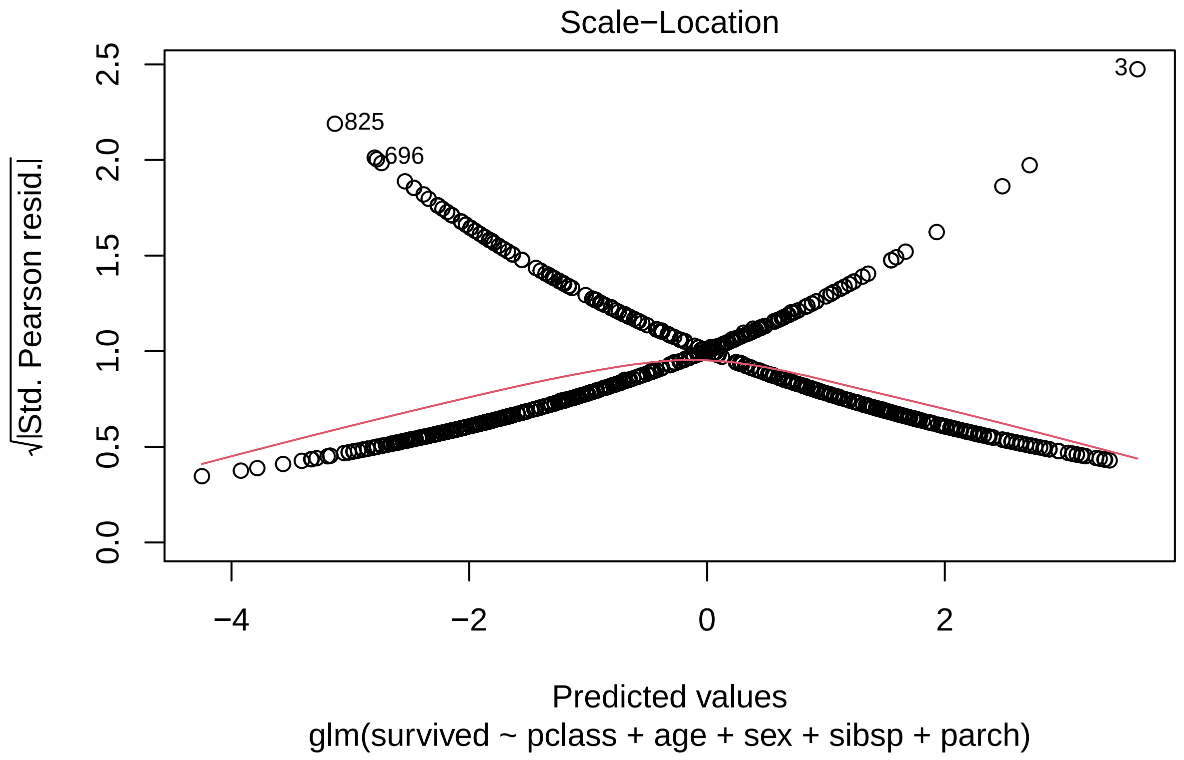
As with Q-Q plots, scale location plots are a challenging to apply to the GLM context.

Observe that the Binomial model actually *assumes* heteroskedasticity will exist in the data:

$$\mathrm{Var}[y_i] = n_i \pi_i (1 - \pi_i)$$

So predicted variance will be maximized when $\hat{\pi}_i = .5$, and minimized at either $\hat{\pi}_i = 0, 1$. It's not hard to convice yourself that this is equivalent to occurring at $\vec{x}_i \hat{\beta} = \mathrm{logit}^{-1}(\hat{\pi}_i) = 0$

```
plot(mod,which=3)
```



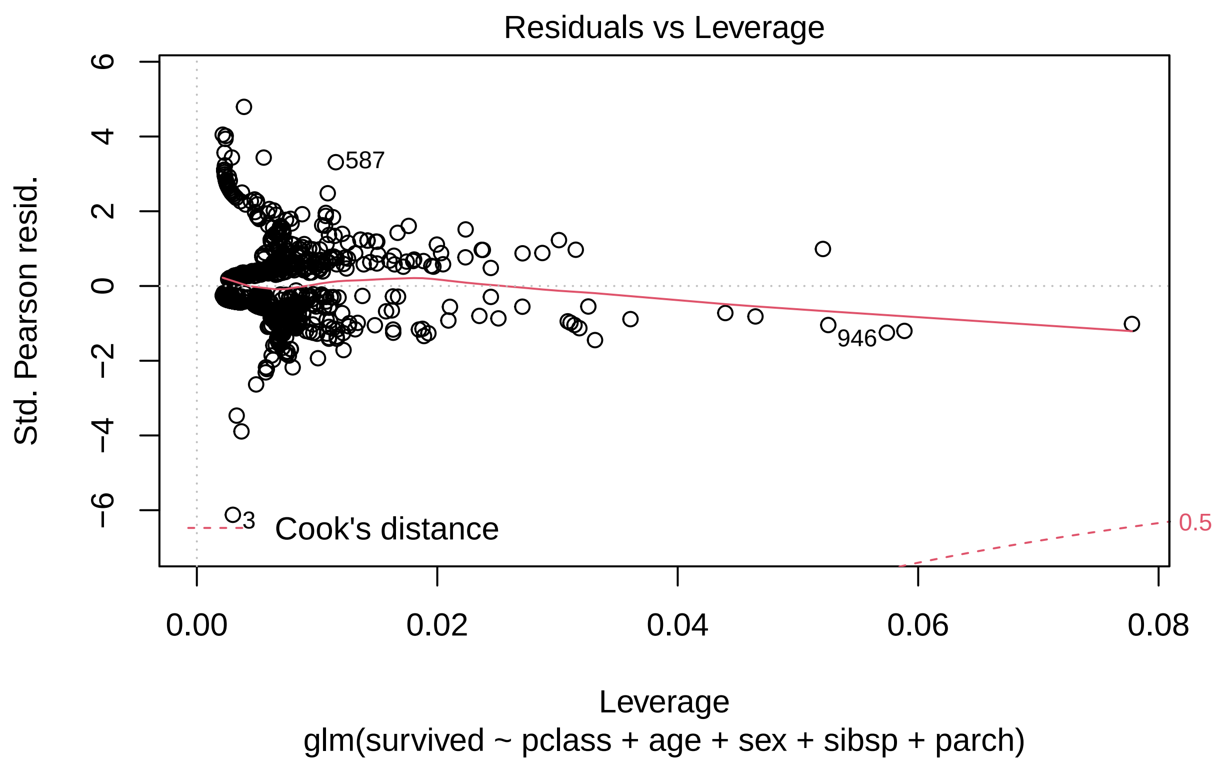
Leverage, Outliers, and Cook's D

As with linear regression, we care a lot about influential observations. We might remove (or model separately) data points with unusually high residual values (deviance or Pearson). We also can still think about *leverage* (h_i). It also turns out that we can still use Cook's D (which I'll denote with a C here to avoid mixing up with deviance):

$$C_i = r_i^{\chi} \frac{h_i}{(1 - h_i)p}$$

Where p is the number of variables in our model. One thing to note here, is is that this version of Cook's D is only an approximation of the average change in predictions, due to the removal of datapoint i. Only for linear regression is this formula exact (Pregibon, 1981).

```
plot(mod,which=5)
```

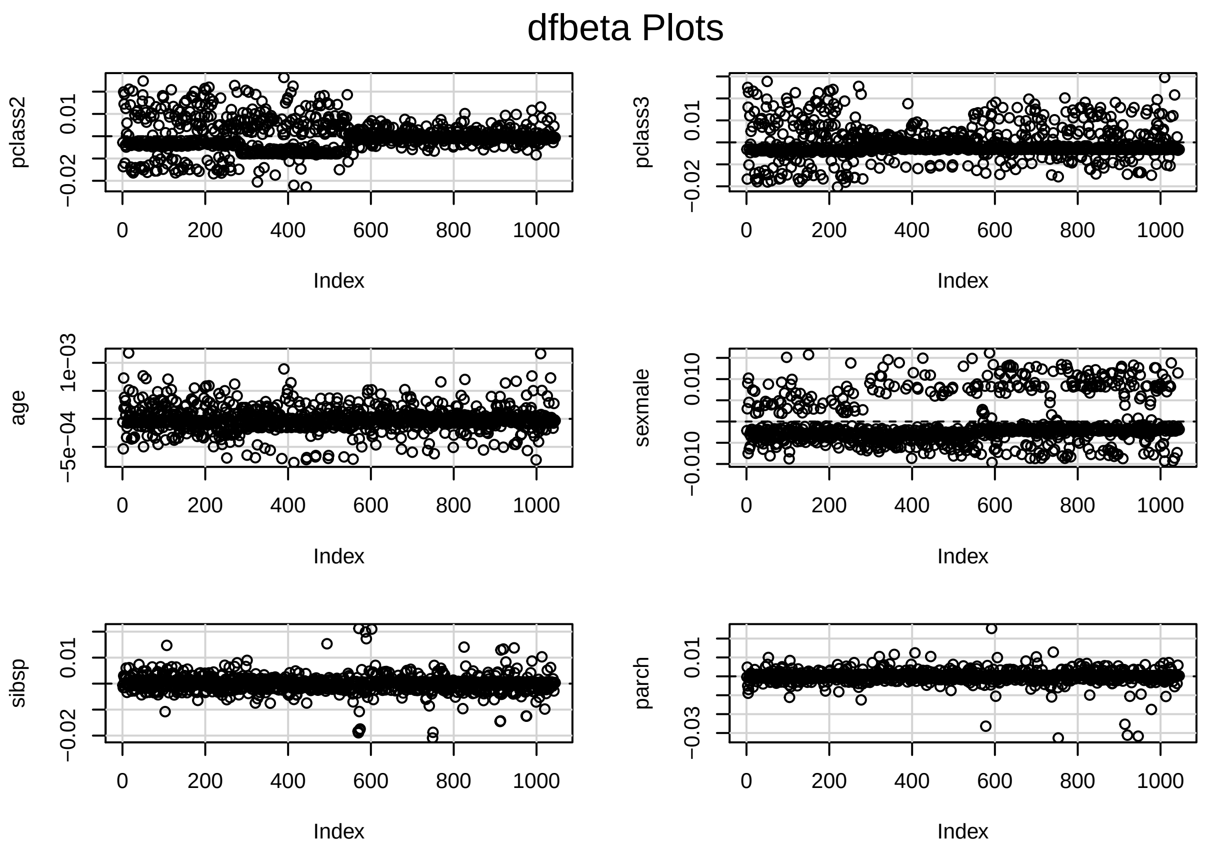


Another measure of influence discussed in IGLM is the "delta-betas", which is more commonly referred to as the DFBETA. Whereas Cook's distance attempts to estimate $\frac{1}{n}\sum_j(\hat{y}_j - \hat{y}_{j,(i)})$ (where $\hat{y}_{j,(i)}$ is the prediction made without the i^{th} datapoint), the DFBETA is $\hat{\beta}_j - \hat{\beta}_{j,(i)}$. In a model with n observations and p variables, there will be $n \times p$ DFBETA values:

```
head(dfbeta(mod))
```

```
##    (Intercept)     pclass2      pclass3           age      sexmale
## 1  0.006156687 -0.00282299 -0.003208433 -5.721998e-05 -0.002151118
## 2  0.025849478 -0.01366594 -0.016687956 -5.354221e-04  0.003010167
## 3 -0.045340297  0.01985923  0.025014566  7.283257e-04  0.009000776
## 4 -0.011976040  0.01446005  0.015999912  1.909365e-04 -0.007510394
## 5 -0.034166363  0.01886647  0.022775838  3.848038e-04  0.010223299
## 6  0.004876234 -0.01210410 -0.010255670  1.905684e-04  0.004520984
##           sibsp         parch
## 1 -0.0006357700 -0.0003716252
## 2 -0.0015241553  0.0048612556
## 3  0.0030269510 -0.0050325042
## 4  0.0001673701 -0.0089520787
## 5  0.0020685926 -0.0069476742
## 6 -0.0015442631 -0.0007589026
```

```
car::dfbetaPlots(mod)
```



For the case of linear regression, averaging the DFBETA over all predictors gives Cook's D (see IGLM 6.2.7).

Overdispersion

One possible difficulty with logistic regression (which we will also see occur in Poisson regression) is *overdispersion*.

Recall that the variance of a Binomial distribution is:

$$\text{Var}[y_i] = n_i\pi_i(1-\pi_i)$$

Overdispersion occurs if the actually observed variance of y_i is greater than this. There are a number of ways to deal with this, but one interesting option is to force your model to account for this overdispersion by assuming:

$$\text{Var}[y_i] = n_i\pi_i(1-\pi_i)\phi$$

Here ϕ is known as the **overdispersion parameter**. This can be fit using R's `quasibinomial` GLM family:

```
mod.logit.overdisp = glm(lhs~dose.dat$dose, family=quasibinomial)
summary(mod.logit.overdisp)
```

```
##
## Call:
## glm(formula = lhs ~ dose.dat$dose, family = quasibinomial)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8986  -0.5475   0.9842   1.3315   1.7179
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -60.103      7.338  -8.191 0.000178 ***
## dose.dat$dose   33.934      4.125   8.227 0.000174 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasibinomial family taken to be 2.018894)
##
##     Null deviance: 284.202  on 7  degrees of freedom
## Residual deviance:  13.633  on 6  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

We see that $\phi \approx 2$ (with $\phi = 1$ just returning the binomial model). In this case it would appear that some overdispersion is possibly present, although more likely this is due to the issue we've seen with the logistic link function:

```
mod.cll.overdisp = glm(lhs~dose.dat$dose, family=quasibinomial(link='cloglog'))
summary(mod.cll.overdisp)
```

```
##
## Call:
## glm(formula = lhs ~ dose.dat$dose, family = quasibinomial(link = "cloglog"))
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.37517  -0.36801   0.07958   0.54314   1.46367
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -38.889      2.969  -13.10 1.22e-05 ***
## dose.dat$dose   21.664      1.648   13.14 1.20e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasibinomial family taken to be 0.9013763)
##
##     Null deviance: 284.2024  on 7  degrees of freedom
## Residual deviance:   5.6281  on 6  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

Now, the "quasibinomial model" is a little weird, because it turns out that we're not using a true "likelihood" when we include the overdispersion parameter. Because of this, some quantities (such as AIC or confidence intervals on ϕ) will not be available to us.

Thus the use of a quasi-binomial model is going to be largely "at your discretion". See IGLM 7.7 for a quick discussion of overdispersion, and a reference to further details if you're interested.

Nominal (Multinomial) Regression

Sometimes it will be the case that your response variable y_i can take on multiple categorical values. If there is no natural ordering to these values, then one method for modeling this data is **multinomial regression**.

The multinomial distribution generalizes the binomial distribution to the case of categorical values. Say that we have k categories, and represent the number of occurrences of each category as $(x_1, ..., x_k)$. The multinomial PMF is then:

$$P[x_1, ..., x_k] = \frac{n!}{x_1!...x_k!}\pi_1^{x_1}...\pi_k^{x_k}$$

Where $n = \sum_j x_j$. It's not hard to convince yourself that when $k = 2$ this reduces to the binomial PMF.

In order to model multinomial data, we first must pick a **reference class**. Since the category values are unordered, it makes no difference which we pick. It's therefore common to choose x_1.

Having chosen our reference class, we now apply a logistic model to the log-odds between each category and the reference:

$$\log(\frac{\pi_j}{\pi_1}) = \vec{x}^T\vec{\beta}_j$$

Notice that this means that, if we have p independent variables, then our final model will have $p \times k$ coefficients (k vectors $\vec{\beta}_j$, each containing p elements).

Since we must have that $\pi_1 + ... + \pi_k = 1$, it can be shown that:

$$\pi_1 = \frac{1}{1 + \exp[\vec{x}^T\vec{\beta}_1]}$$
$$\pi_{j=2,...,k} = \frac{\exp[\vec{x}^T\vec{\beta}_j]}{1 + \exp[\vec{x}^T\vec{\beta}_j]}$$

As with binomial logistic regression, we can interpret the coefficients in terms of odds ratios. Say we have a single independent variable x, the odds ratio for an individual with with variable value $x+1$ vs an individual with variable x is:

$$\text{OR}_j = \frac{\pi_{j,x+1}}{\pi_{1,x+1}} / \frac{\pi_{j,x}}{\pi_{1,x}} = \exp\beta_{1j}$$

Let's apply a multinomial model to model the island distribution of the penguins dataset:

```r
library(nnet) # R doesn't fit multinomial models normally, so we'll need a package
library(palmerpenguins)
dat = penguins %>% drop_na

mod = multinom(island~bill_length_mm + bill_depth_mm + body_mass_g,data=dat)
```

```
## # weights:  15 (8 variable)
## initial  value 365.837892
## iter  10 value 197.285326
## iter  20 value 195.293460
## iter  30 value 195.288608
## final  value 195.288548
## converged
```

```r
summary(mod)
```

```
## Call:
## multinom(formula = island ~ bill_length_mm + bill_depth_mm +
##     body_mass_g, data = dat)
```

```
##
## Coefficients:
##           (Intercept) bill_length_mm bill_depth_mm  body_mass_g
## Dream      -10.575406     0.22709868     0.6861920 -0.002801124
## Torgersen   -5.873156    -0.03827875     0.7668068 -0.001778630
##
## Std. Errors:
##           (Intercept) bill_length_mm bill_depth_mm  body_mass_g
## Dream     0.002104397     0.04224210    0.06594702 0.0003931631
## Torgersen 0.003726580     0.05222746    0.10248151 0.0005004206
##
## Residual Deviance: 390.5771
## AIC: 406.5771
```

```
confint(mod)
```

```
## , , Dream
##
##                       2.5 %        97.5 %
## (Intercept)    -10.579530607 -10.571281521
## bill_length_mm   0.144305692   0.309891672
## bill_depth_mm    0.556938202   0.815445782
## body_mass_g     -0.003571709  -0.002030538
##
## , , Torgersen
##
##                       2.5 %        97.5 %
## (Intercept)    -5.880459486 -5.8658515602
## bill_length_mm -0.140642693  0.0640851977
## bill_depth_mm   0.565946755  0.9676668782
## body_mass_g    -0.002759436 -0.0007978235
```