Playing Around with Model Residuals

Sam Mason

6/8/2021

Response Residuals

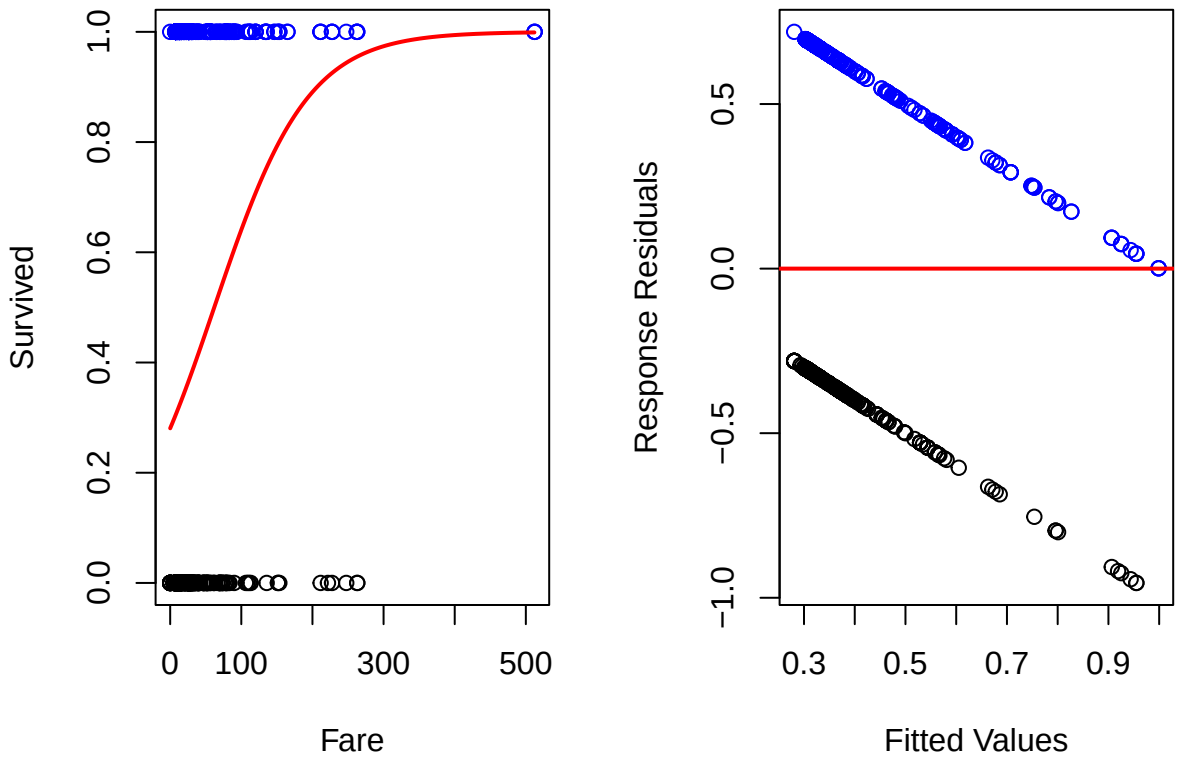
```
titanic <- read.csv("/Users/sam/Documents/Teaching/Data Science Practicum/Lecture Materials/Matrices & I
mod <- glm(Survived ~ Fare, data = titanic, family = binomial(link = "logit"))
```

For a binomial GLM with a binary response (where all observations are modeled as random variables described by indepenedent Bernoulli distributions), the response residuals can be defined as

$$r_i = y_i - \hat{p}_i$$

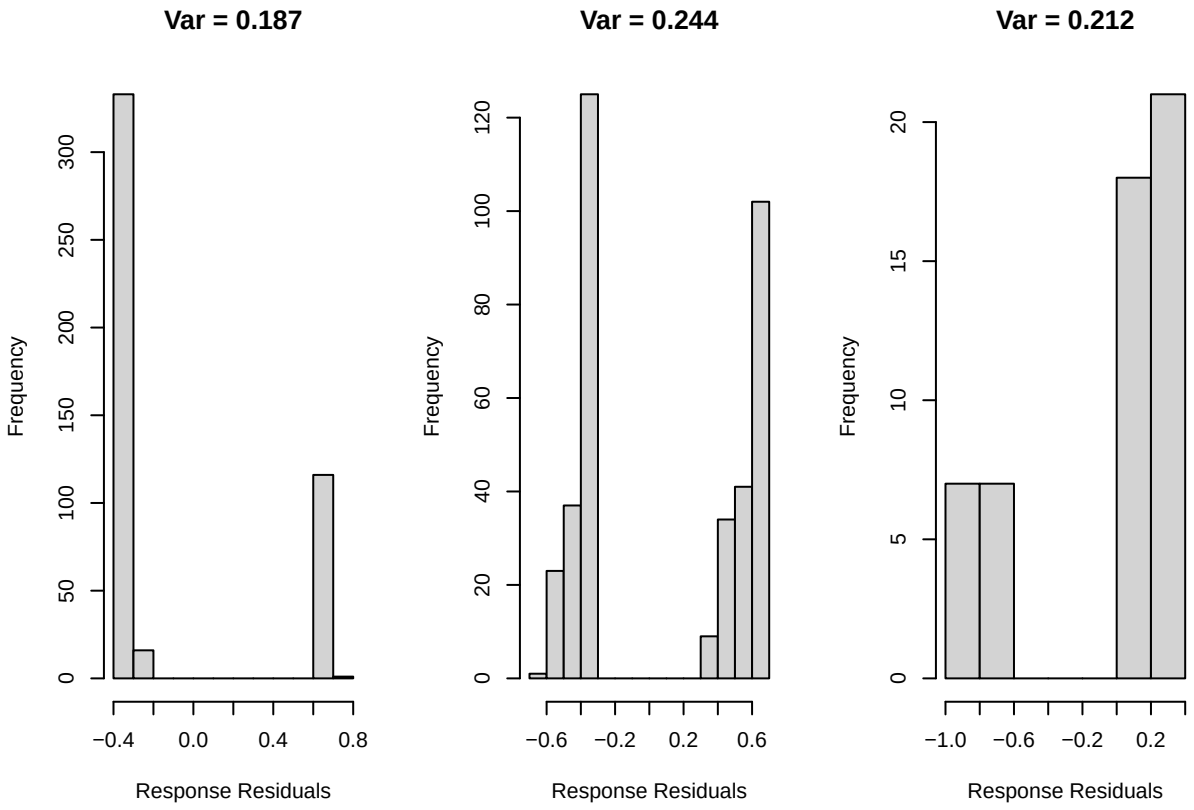
Response residuals give you information about your model error (a broad term that I use to mean, "the degree to which your model fails to explain the observed variation in the response variable"), but are often not particularly helpful in a GLM-context. To understand why, let's make a few plots here.

```
plot_df <- data.frame(Fare = seq(min(titanic$Fare), max(titanic$Fare), 1))
plot_df$Preds <- predict(mod, newdata = plot_df, type = "response")
fits <- mod$fitted.values
resids <- titanic$Survived - mod$fitted.values
palette(c("black", "blue"))
par(mfrow = c(1, 2), mar = c(5, 4, 2, 2))
plot(Survived ~ Fare, data = titanic, col = as.factor(titanic$Survived))
lines(x = plot_df$Fare, y = plot_df$Preds, lwd = 2, col = "red")
plot(resids ~ fits, xlab = "Fitted Values", ylab = "Response Residuals",
     col = as.factor(titanic$Survived))
abline(a = 0, b = 0, lwd = 2, col = "red")
```



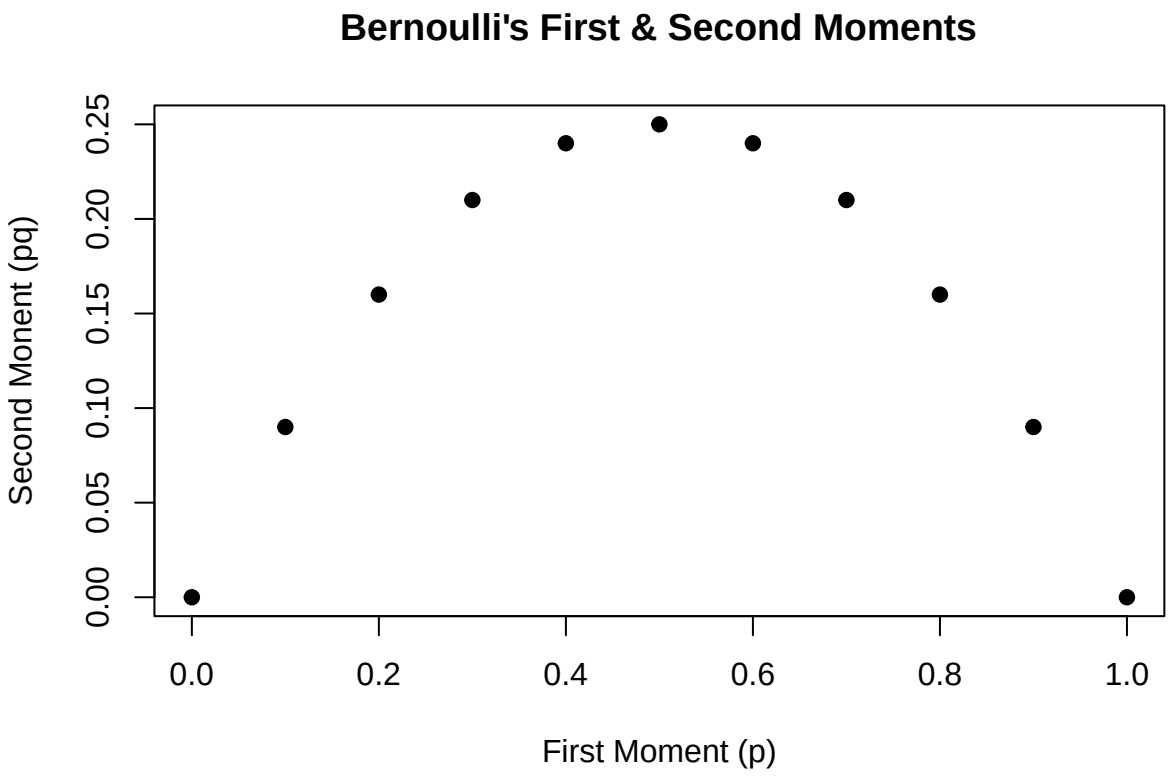
As we can expect, when the probability of survival is low (small values on y-axis of the first plot; small values on the x-axis of the second plot), the model deviates less from observed death (black dots) than it does from observed survival (blue dots). Taking a closer look at the second plot, it looks like our model fits the data best at the extremes of the fitted values, but does a generally poor job at intermediate values of the predicted probability of survival. The goal in modeling is to parameterize a model that fits all of your data, not just the extreme. Does this mean the model we've fit is bad? Let's look next at the distribution of response residuals at different locations along the fitted values.

```
var_1 <- as.character(round(var(resids[fits < 0.33]), 3))
var_2 <- as.character(round(var(resids[fits >= 0.33 & fits <= 0.66]), 3))
var_3 <- as.character(round(var(resids[fits > 0.66]), 3))
par(mfrow = c(1, 3))
hist(resids[fits < 0.33], main = paste("Var =", var_1, sep = " "),
     xlab = "Response Residuals")
hist(resids[fits >= 0.33 & fits <= 0.66], main = paste("Var =", var_2, sep = " "),
     xlab = "Response Residuals")
hist(resids[fits > 0.66], main = paste("Var =", var_3, sep = " "),
     xlab = "Response Residuals")
```



The first histogram describes the response residuals corresponding to fitted values less than 0.33, the second to fitted values in the range [0.33, 0.66], and the third to fitted values greater than 0.66. The trend is rough because our data is messy, and I've only broken the residuals up into three groups, but take a look at the calculated sample variances above each histogram. Take a moment to think about why those values broadly agree with the familiar plot I show below.

```
m1 <- seq(0, 1, 0.1)
q <- 1-m1
m2 <- m1*q
plot(m1, m2, pch=19, xlab = 'First Moment (p)', ylab = 'Second Monent (pq)',
     main = "Bernoulli's First & Second Moments")
```



Okay, so here's the big question: do we have a bad model, or do the response residuals behave as we'd expect them to behave for this type of model?

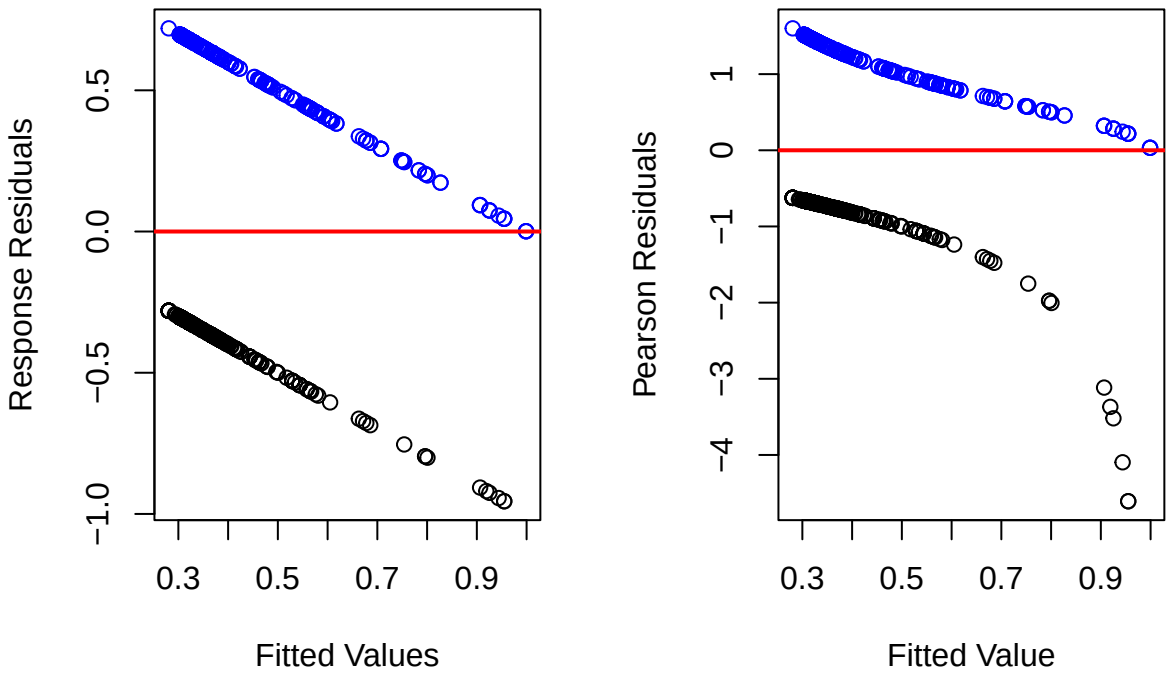
Pearson Residuals

For a binomial GLM with a binary response (where all observations are modeled as random variables described by independent Bernoulli distributions), the Pearson residuals can be defined as

$$r_i^p = \frac{y_i - n_i\hat{p}_i}{\sqrt{n_i\hat{p}_i(1-\hat{p}_i)}}$$

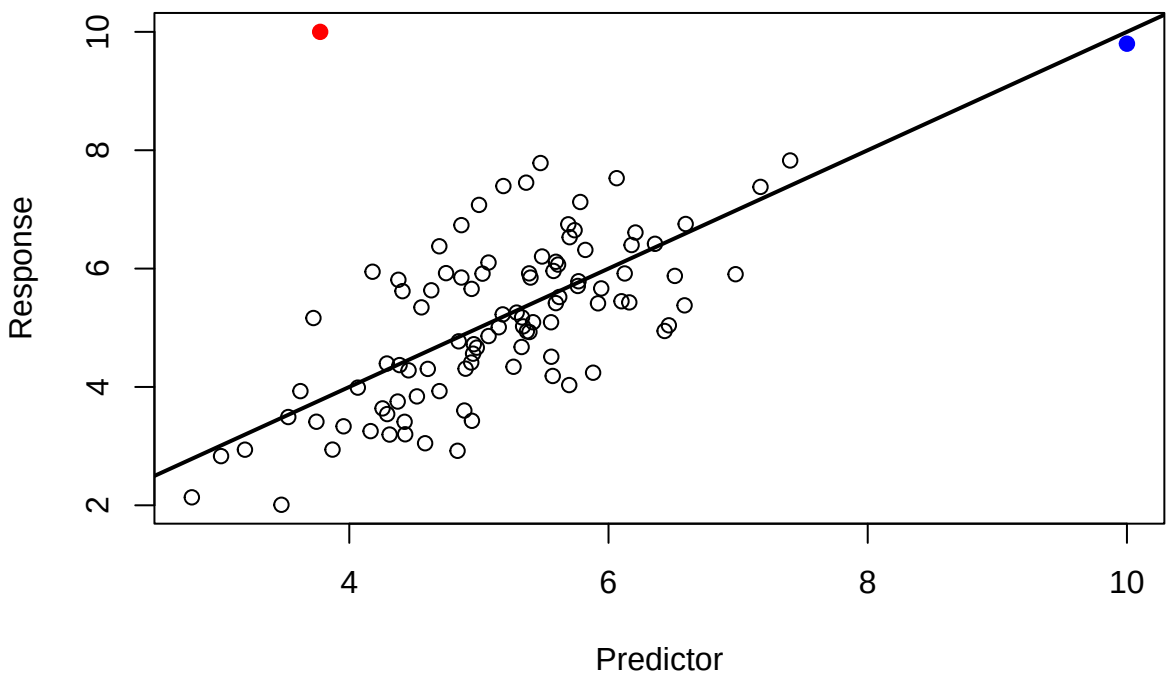
where n_i is the number of trials (again, the Bernoulli distribution is a special case of the binomial where the number of trials is equal to 1). The most important thing that I want you to understand from the equation above is that, at its core, the Pearson residual is just the response residual scaled by the standard deviation of the Bernoulli distribution associated with p_i. Let's see how the two residuals compare graphically.

```
par(mfrow = c(1, 2))
plot(resids ~ fits, xlab = "Fitted Values", ylab = "Response Residuals",
     col = as.factor(titanic$Survived))
abline(a = 0, b = 0, lwd = 2, col = "red")
plot(residuals(mod, type = "pearson") ~ fits, xlab = "Fitted Value",
     ylab = "Pearson Residuals", col = as.factor(titanic$Survived))
abline(a = 0, b = 0, lwd = 2, col = "red")
```



You'll notice that the range of y-axis has increased sharply because we're dividing by decimal standard deviations. The Pearson residuals seem to "pinch" all of the residuals a bit closer to the trend line (red), and, even more importantly, they really highlight how poorly our model fits the zero data (black dots) as the probability of survival approaches 1. This lack of fit wasn't as obvious when looking at a plot of the response residuals.

The Leverage of a Data Point



In the plot above, the blue dot has a lot of leverage, but is not an outlier, while the red dot does not have a lot of leverage, but might be considered an outlier. Leverage is a measure of how close a particular data point is to all the other data points along the x-axis. We can calculate leverage in R using the `hatvalues()` function.

```
hatvalues(lm(y ~ x, data = df))[95:100] # the 100th datapoint is the blue dot
```

```
##         95         96         97         98         99        100
## 0.02963605 0.01151151 0.03010799 0.01526401 0.02868272 0.23675423
```

Standardized Pearson Residuals

For a binomial GLM with a binary response (where all observations are modeled as random variables described by independent Bernoulli distributions), the standardized Pearson residuals can be defined as

$$r_i^{sp} = \frac{r_i^p}{\sqrt{1 - h_i}}$$

where h_i is the leverage of a given data point. Standardized Pearson residuals can then be understood as scaling the Pearson residuals such that points with greater leverage get inflated residuals. Why would we want this?

```
par(mfrow = c(1, 3))
plot(resids ~ fits, xlab = "Fitted Values", ylab = "Response Residuals",
     col = as.factor(titanic$Survived))
abline(a = 0, b = 0, lwd = 2, col = "red")
plot(residuals(mod, type = "pearson") ~ fits, xlab = "Fitted Value",
     ylab = "Pearson Residuals", col = as.factor(titanic$Survived))
abline(a = 0, b = 0, lwd = 2, col = "red")
plot((residuals(mod, type = "pearson")/sqrt(1 - hatvalues(mod))) ~ fits,
     xlab = "Fitted Value", ylab = "Standardized Pearson Residuals",
     col = as.factor(titanic$Survived))
abline(a = 0, b = 0, lwd = 2, col = "red")
```