Lecture 13

Peter Shaffery

2/24/2021

Inference with GLMs

Last lecture we introduced *generlized linear models*, that is models of the form:

$$g(E[y_i]) = \vec{x}_i^T \vec{\beta}$$

Where g is referred to as the *link* function.

Recall that, while these models provided a great deal of flexibility, they came at the cost of no longer being able to find the MLE $\hat{\beta}$ explicitly. Instead, we had to use numerical optimization (ie. the `optim` function) to get an *approximate* MLE.

This leaves us in a bit of bind when it comes to performing inference. For linear regression, we had a closed-form expression for $\hat{\beta}$, so we could derive it's sampling distribution explicitly. Recall that in SLR, we knew that:

$$\hat{\beta}_1 \sim N(\beta_1, \text{s.e}(\beta_1))$$

So we could do things like construct confidence intervals based on our choice of significance α:

$$[\hat{\beta}_1 - t_{(1-\alpha)/2}\text{s.e.}(\beta_1), \hat{\beta}_1 - t_{1-\alpha/2}\text{s.e.}(\beta_1)]$$

Or compute p-values:

$$p = \Pr[t > |\hat{t}| \mid \beta_1 = 0]$$

These were essential tools for performing inference. Can we establish the sampling distribution of $\hat{\beta}$ for a GLM? Yes!

Spoiler alert: it will still be (approximately) normal

The Poisson Distribution

To make this discussion more concrete, let's look at an example. On Tuesday we introduced one of the most common members of exponential family to be used in a GLM. Today we will look at another popular distribution: Poisson.

Poisson distributions model "event counts", the number of time some event of interest occurs in a fixed amount of time. For example, the number of buses which arrive a specific bus stop in any given hour might be modeled as Poisson random variable.

Poisson random variables are *discrete*, and their PMF is given:

$$f(k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Where the single parameter λ is referred to as the "rate" (as in, the rate of events occurring per unit time). We must have that $\lambda > 0$.

If k is Poisson then $E[k] = \text{Var}[k] = \lambda$ (this is a weird property that Poissons have, which we will absolutely come back to when we do Poisson regression).

As you might expect, the Poisson is a member of the exponential family, let's confirm that now:

$$\begin{aligned} f(k;\lambda) &= \frac{\lambda^k e^{-\lambda}}{k!} \\ &= \exp\left[k\log\lambda - \lambda - \log k!\right] \\ &= \exp\left[a(k)b(\lambda) + c(\lambda) + d(k)\right] \end{aligned}$$

Observing that since $a(k) = k$ the exponential form is the *canonical form* and $b(\lambda) = \log\lambda$ is the natural parameter.

Example: Warp Breaks

Say that we are (particularly stats literate) weavers, making textiles on a loom. Often, when we finished a piece of textile, the woven thread may break in multiple places (a *warp break*). This is undesirable as it lowers the finished value of the textile.

Say that we've recorded the number of warp breaks for 57 textiles that we've made in the past. We would like to model the number of warp breaks which occur in a given textile as a Poisson random variable. Let's compute the MLE $\hat{\lambda}$ given our data.

```
library(tidyverse)
library(magrittr)
library(ggplot2)
library(datasets)

dat = warpbreaks
dat %>% head
```

```
##   breaks wool tension
## 1     26    A       L
## 2     30    A       L
## 3     54    A       L
## 4     25    A       L
## 5     70    A       L
## 6     52    A       L
```

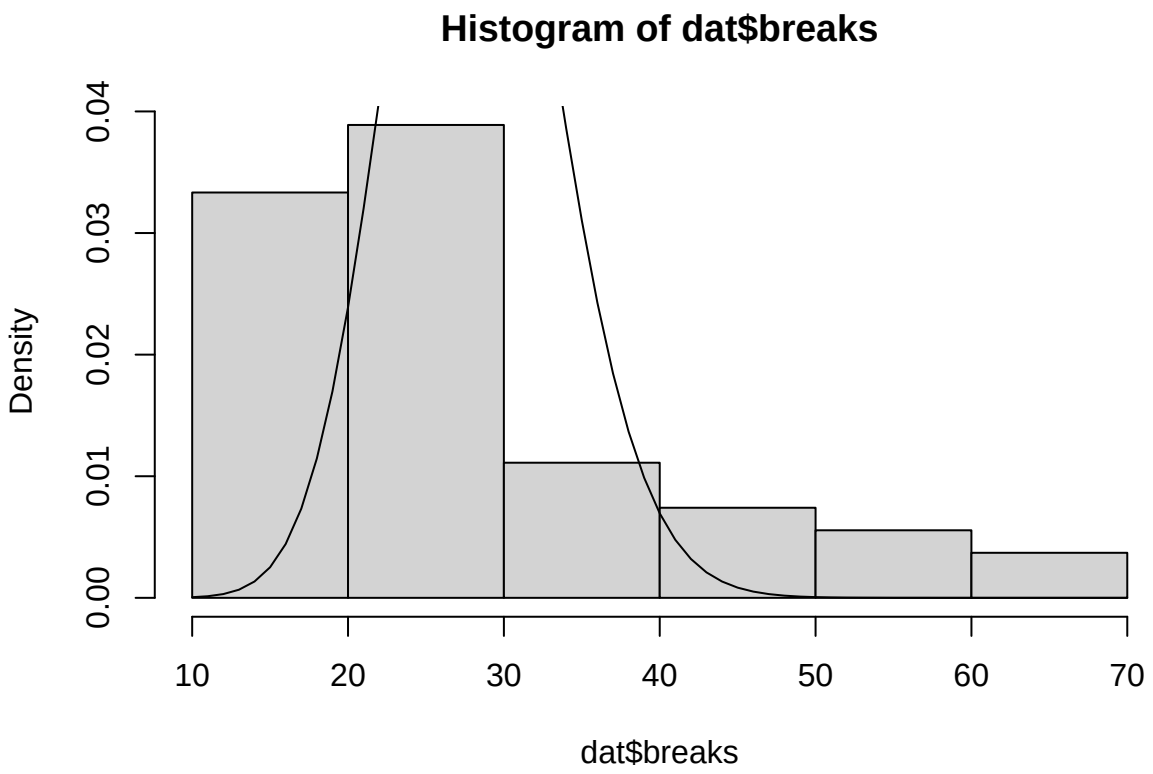
Now, it's actually fairly straightforward to compute the MLE $\hat{\lambda}$ by hand (and it's a good exercise to try out). But for illustration purposes, we will use `optim` instead:

```
nll = function(l,x){
  if (l<0){
    return(1e6) # since the likelihood when lambda=0 is also 0, and log(0) is undefined, instead just a
  }

  else{
    return(-sum(dpois(x,l,log=TRUE)))
  }
}
first.guess = c(1)
mle = optim(first.guess,
            nll,
            x=dat$breaks,
```

```
            method='BFGS') #optim will whine about 1-D if you don't use BFGS

xgrid = seq(10,max(dat$breaks),by=1)
pmf = dpois(xgrid,mle$par)
hist(dat$breaks, freq=FALSE)
lines(xgrid,pmf)
```



Now, let's say that we would like to examine how the textile material changes the number of warp breaks which occur. We have two types of wool in our dataset, type A and type B. We will therefore use a model of the form:

$$E[\text{BREAKS}_i] = \exp(\beta_A + \beta_B \text{WOOL}_i)$$

Where WOOL_i is a dummy variable which is 0 when the i^{th} observation is of type A and 1 when it is of type B.

Note here that we are using an exponent so that $E[\text{BREAKS}_i] = \lambda_i$ will always be positive. Our link function here is therefore $g(x) = \log(x)$.

We would like to determine whether $\beta_B = 0$ or not, that is if wool type B is significantly different from type A.

To do so we will need derive the sampling distribution for $\vec{\beta} = [\beta_A, \beta_B]^T$. Once we have that, we can compute a confidence interval (CI) for β_B and make a decision about significance.

\begin{Math}

It turns out that the easiest way to get a sampling distribution for $\vec{\beta}$ is to go through the *score function*.

Recall from last lecture that we defined the score function as the first-derivative of the log-likelihood function. When the parameter is a *vector* then so is the output of the score function:

$$\begin{aligned}\vec{U}(\beta_A, \beta_B) &= [\frac{d}{d\beta_A} l(\beta_A, \beta_B), \frac{d}{d\beta_B} l(\beta_A, \beta_B)]^T \\ &= [U_A(\beta_A, \beta_B), U_B(\beta_A, \beta_B)]^T\end{aligned}$$

However for simplicity, we'll just focus on U_B and treat $\hat{\beta_A}$ as if it were a known constant, so U_B will only depend on β_B.

Recall from Calc 2 that we can *approximate* any function, with a truncated series of polynomials (a Taylor Series):

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2}(x-a)^2 + ...$$

We can use a Taylor series here to approximate $U(\beta_B)$ to first order, around $a = \hat{\beta}_B$. This gives us that:

$$U_B(\beta_B) \approx U(\hat{\beta}_B) + U'_B(\hat{\beta}_B)(\beta_B - \hat{\beta}_B)$$

Since, by definition, $U(\hat{\beta_B}) = 0$, where therefore have:

$$U_B(\beta_B) \approx U'_B(\hat{\beta}_B)(\beta_B - \hat{\beta}_B)$$

And thus, finally:

$$(\beta_B - \hat{\beta}_B) \approx \frac{U_B(\hat{\beta_B})}{U'_B(\hat{\beta}_B)}$$

We are almost done, but to make things easier on ourselves we will make one more approximation to get rid of U'_B.

First, observe that we can write the score function as a *sum* of individual data points' score functions:

$$\begin{aligned}U(\theta) &= \frac{d}{d\theta} l(\theta; y_1, ..., y_n) \\ &= \frac{d}{d\theta} \sum_{i=1}^{n} l(\theta; y_i) \\ &= \sum_{i=1}^{n} \frac{d}{d\theta} l(\theta; y_i) \\ &= \sum_{i=1}^{n} U(\theta)^{(i)}\end{aligned}$$

The same thing is true for $U'_B(\beta_B)$:

$$U'_B = \sum_{i=1}^{n} U_B^{'(i)}$$

This suggests to us that can get rid of U'_B by approximating it with $nE[U_B^{'(i)}]$ (since for large n $\frac{1}{n}\sum_{i=1}^{n} U_B^{'(i)} \approx E[U_B^{'(i)}]$)

$$(\beta_B - \hat{\beta}_B) \approx \frac{U_B(\beta_B)}{nE[U_B^{'(i)}]}$$

We'll denote this expected value:

$$J = -E[U_B^{'(i)}]$$

The negative pops up for a good reason, which we'll see in just a second, but for now just think of it as a way to turn $(\beta_B - \hat{\beta_B})$ into $(\hat{\beta}_B - \beta_B)$ making our lives easier. Now, let's also move that n in the denominator up to the numerator:

$$(\hat{\beta}_B - \beta_B) \approx \frac{(1/n)U_B}{J}$$

We are now in a position to prove our final result. By subbing the true value of β_B, β_B^*, into the above expression we will get that for large n, $\hat{\beta}_B$ is approximately normal:

$$\hat{\beta}_B \to N(\beta_B^*, J/n)$$

This result relies on three important properties of the score function U_B:

1. $E[U_B] = 0$
2. $\text{Var}[U_B] = J$
3. When sample size n is large, then $\frac{1}{n}U_B \to N(0, J/n)$

Let's look at why these are true.

Property 1: $E[U_B] = 0$

This falls out of a fact that we talked about last lecture:

$$\int \frac{d}{d\theta} f(y;\theta) dy = 0$$

Observe that (for a single data point) we use the chain rule to rewrite $U(\theta)$:

$$U(\theta) = \frac{d}{d\theta} \log f(y;\theta) = \frac{1}{f(y;\theta)} \frac{d}{d\theta} f(y;\theta)$$

Therefore:

$$\begin{aligned} E[U(\theta)] &= \int U(\theta) f(y;\theta) dy \\ &= \int \frac{1}{f(y;\theta)} \frac{d}{d\theta} f(y;\theta) f(y;\theta) dy \\ &= \int \frac{d}{d\theta} f(y;\theta) dy \\ &= 0 \end{aligned}$$

Property 2: $\textbf{Var}[U_B] = 0$

Start with the fact that:

$$\text{Var}[U_B] = E[U_B^2] - E[U_B]^2$$

We just showed that $E[U_B] = 0$, thus:

$$\text{Var}[U_B] = E[U_B^2]$$

Now, a weird fact about U_B is that $-U_B' = U_B^2$, and therefore:

$$\text{Var}[U_B] = E[U_B^2] = -E[U_B'] = J$$

For a proof of this see IGLM page 50.

In IGLM the quantity J is referred to as the **information**. A more common definition of information is actually $1/J$ (since this way low information implies a high variance for $\hat{\beta_B}$).

Property 3: $\frac{1}{n}U_B \to N(0, J/n)$

We've already seen that:

$$U_B = \sum_{i=1}^{n} U_B^{(i)}$$

This means that:

$$\frac{1}{n}U_B = \frac{1}{n}\sum_{i=1}^{n} U_B^{(i)}$$

Has the form of a *sample mean*. For series of iid random variables X_i (where $\text{Var}[X] < \infty$), it is true that:

$$\frac{1}{n}\sum_{i=1}^{n} X_i \to N(E[X_i], \text{Var}[X_i]/n)$$

This is known as the **Central Limit Theorem** (CLT), and is one of the most important results in statistics and probability.

Applying the CLT to $U_B^{(i)}$, and pulling in Properties 1 and 2 to get $E[U_B^{(i)}] = 0$ and $\text{Var}[U_B^{(i)}] = J$ gives us the desired result:

$$\frac{1}{n}U_B \to N(0, J/n)$$

\end{Math}

Alright! We have gotten our desired sampling distribution for $\hat{\beta}_B$, when we have a large number of data points (typically said to be $n > 30$) we have:

$$\hat{\beta}_B \to N(\beta_B^*, J/n)$$

Note that when we are performing linear regression this approximation is exact.

For most problems, we will have a little more work to do before we can get s.e.($\hat{\beta}_B$). Specifically, we'll need to estimate J before we can compute s.e($\hat{\beta}_B$, and this will makes the math a little more gross. So today we will just let R do all the heavy lifting for us.

Example: Warp Breaks (con't)

R provides a special `glm` function, which will allow us to fit and analyze a generalized linear model in the same way that we did `lm`, but first, let's quickly look at how we could fit this model ourselves:

First, we need to fit our model:

```
nll = function(b,breaks,wool){
  bA = b[1]
  bB = b[2]
  lam = exp(bA + bB*wool)
```

```
  terms = dpois(breaks,lam,log=TRUE)

  return(-sum(terms))
}

first.guess = c(0,0)
dat$wool.ind = (dat$wool=='B')
mle = optim(first.guess,
            nll,
            breaks=dat$breaks,
            wool=dat$wool.ind)
mle
```

```
## $par
## [1]  3.4351359 -0.2060799
##
## $value
## [1] 277.9988
##
## $counts
## function gradient
##       81       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

Now let's see what R gets:

```
mod = glm(breaks~wool, family=poisson, data=dat)
mod %>% summary
```

```
##
## Call:
## glm(formula = breaks ~ wool, family = poisson, data = dat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.4071  -1.9148  -0.7138   0.8332   5.9948
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.43518    0.03454  99.443  < 2e-16 ***
## woolB       -0.20599    0.05157  -3.994 6.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 297.37  on 53  degrees of freedom
## Residual deviance: 281.33  on 52  degrees of freedom
## AIC: 560
##
```

```
## Number of Fisher Scoring iterations: 4
```

We see that the our MLE estimate is exactly what R got. Furthermore, we see that R has computed the standard error of β_B for us, as well as a p-value. We would interpret this result as indicating that wool type B does reduce the overall indicidence of warp breaks.

We'll talk more about how we can interpret the rest of this table starting next Tuesday when we get into logistic regression.