MA478 - Lesson 5

Clark

Today we're going to talk about fitting GLMs. That is, finding β values that best project some function of our mean onto the column space of our design matrix.

To get there, I first want to backtrack just a little and talk about weighted least squares. If we recall, we could rewrite our linear regression model as:

A more general structure would be to allow $Cov(y) = \sigma^2\mathbf{V}$. If we KNOW $\mathbf{V}$, then we could use this to rewrite X and y as:

The reason for this, is now the relationship between $\mathbf{X^*}$ and $\mathbf{y^*}$ follows our typical assumptions for linear regression. That is we have:

Therefore, we can find the least squares estimates of β as:

Our estimate of β is sometimes called the GLS estimate, or the *generalized least squares* estimate of β. If we assume that $\mathbf{V}$ is diagonal $var(y_i) = \frac{\sigma^2}{w_i}$, our estimates of β give more weight to some observations than to others, then we call the estimates the weighted least squares estimator.

From Fitting LMs to Fitting GLMs

Unfortunately unlike fitting a linear model, we can't directly minimize the SSE. Most often in order to estimate β we have to rely on maximum likelihood estimation. Just as a quick aside, in case you haven't had MA476, MLEs are found in the following way:

Therefore, most often we don't want to maximize the likelihood, instead we want to maximize the log likelihood. For a single observation from a GLM (note this is true for ANY GLM) we can write:

Now note that we want to maximize this for β but there aren't any β terms! So, we have to have our cascading functions:

So, to take the derivative of a function inside of a function (inside of a function, inside of a function) we have to rely on our old friend the chain rule.

For what seems like no reason, we're going to rewrite this in the following way:

If, instead, we had started out with the weighted least squares criterion:

and we attempted to minimize it (WRT β) then we ALSO would have ended up with the same thing!

This suggests (and I'm skipping a TON of steps here that Agresti goes through but I don't think they're helpful at this stage) that we can find estimates of β by minimizing a weighted least squares type equation. The complete algorithm is:

So, we need to know $\eta = g(\mu)$ and $V(\mu)$, but we really don't need any further information about the distribution of $\mathbf{y}$ other than the fact that it is part of the exponential dispersion family.

Let's put this all together for a model where we assume a Poisson distribution with a log link.

```
# Example: this data set gives the number of warp breaks per loom, where a loom
# corresponds to a fixed length of yarn.
library(faraway)
```

```
## Warning: package 'faraway' was built under R version 4.3.2
```

```
X <- model.matrix(~wool*tension,
                  data=warpbreaks) # wool, tension and interaction variables
y <- warpbreaks$breaks # the number of breaks (count response)


mu <- y
eta <- log(mu)
z <- eta + (y-mu)/mu
w <- mu
```

```
lmod <- lm(z~wool*tension,weights=w,warpbreaks)
coef(lmod)
```

```
##    (Intercept)          woolB       tensionM       tensionH woolB:tensionM
##      3.8713986     -0.4764736     -0.6348096     -0.5925204      0.6477783
## woolB:tensionH
##      0.1593500
```

```
for(i in 1:5){
  eta <- lmod$fitted.values
  mu <- exp(eta)
  z <- eta + (y-mu)/mu
  w <- mu
  lmod <- lm(z~wool*tension,weights=w,warpbreaks)
  cat(i, coef(lmod), "\n")
}
```

```
## 1 3.799456 -0.457871 -0.6197219 -0.5955579 0.6388878 0.1870658
## 2 3.796741 -0.4566298 -0.6186853 -0.595798 0.6381787 0.1883615
## 3 3.796737 -0.4566272 -0.618683 -0.5957987 0.6381768 0.1883632
## 4 3.796737 -0.4566272 -0.618683 -0.5957987 0.6381768 0.1883632
## 5 3.796737 -0.4566272 -0.618683 -0.5957987 0.6381768 0.1883632
```

Now, the one thing this doesn't give us is the standard error of $\boldsymbol{\beta}$.

To solve for this we note that we can estimate the variance through:

For a Poisson regression, $\phi = 1$ so all we need are the weights from above.

```
wm <- diag(w)
sqrt(diag(solve(t(X)%*%wm%*%X)))
```

```
##    (Intercept)          woolB       tensionM       tensionH woolB:tensionM
##     0.04993762     0.08019210     0.08440021     0.08377740     0.12215321
## woolB:tensionH
##     0.12989542
```

Which we can compare to:

```
r_glm = glm(breaks~wool*tension,data=warpbreaks,
            family=poisson(link="log"))

summary(r_glm)
```

```
##
## Call:
```

```
## glm(formula = breaks ~ wool * tension, family = poisson(link = "log"),
##     data = warpbreaks)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     3.79674    0.04994  76.030  < 2e-16 ***
## woolB          -0.45663    0.08019  -5.694 1.24e-08 ***
## tensionM       -0.61868    0.08440  -7.330 2.30e-13 ***
## tensionH       -0.59580    0.08378  -7.112 1.15e-12 ***
## woolB:tensionM  0.63818    0.12215   5.224 1.75e-07 ***
## woolB:tensionH  0.18836    0.12990   1.450    0.147
## ---
## 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: 182.31  on 48  degrees of freedom
## AIC: 468.97
##
## Number of Fisher Scoring iterations: 4
```

Let's say, instead, we wanted to use the Gamma distribution for Y. How does this differ? What would we have to change in our setup? What choices do we have to make?