4.4 GLM: the Poisson Regression Model

Since in the Pima data (Section 4.3.2) the number of pregnancies is a count, we might consider predicting it using Poisson regression.[7] Here's how we can do this with **glm()**:

```
> poisout <- glm(NPreg ~ .,data=pima,family=poisson)
> summary(poisout)
...
Coefficients:
             Estimate Std. Error z value
(Intercept)  0.2963661  0.1207149   2.455
Gluc        -0.0015080  0.0006704  -2.249
BP           0.0011986  0.0010512   1.140
Thick        0.0000732  0.0013281   0.055
Insul       -0.0003745  0.0001894  -1.977
BMI         -0.0002781  0.0027335  -0.102
Genet       -0.1664164  0.0606364  -2.744
Age          0.0319994  0.0014650  21.843
Diab         0.2931233  0.0429765   6.821
            Pr(>|z|)
(Intercept)  0.01408 *
Gluc         0.02450 *
BP           0.25419
Thick        0.95604
Insul        0.04801 *
BMI          0.91896
Genet        0.00606 **
Age          < 2e-16 ***
Diab        9.07e-12 ***
...
```

On the other hand, even if we believe that our count data follow a Poisson distribution, there is no law dictating that we use Poisson regression, i.e., the model (4.10). As mentioned following that equation, the main motivation for using exp() in that model is to ensure that our regression function is nonnegative, conforming to the nonnegative nature of Poisson random variables. This is not unreasonable, but as noted in a somewhat different context in Section 3.3.7, transformations — in this case, the use of exp() — can produce distortions. Let's try the "unorthodox" model, (4.11):

[7]It may seem unnatural to predict this, but as noted before, predicting any variable may be useful if data on that variable may be missing.

```
> quasiout <- glm(NPreg ~ .,data=pima,
     family=quasi(variance="mu^2"),start=rep(1,9))
```

This “quasi” family is a catch-all option, specifying a linear model but here allowing us to specify a Poisson variance function:

$$Var(Y \mid X = t) = [\mu(t)]^2 \tag{4.39}$$

with $\mu(t) = t'\beta$. This is (4.11), not the standard Poisson regression model, but worth trying anyway.

Well, then, which model performed better? As a rough, quick look, ignoring issues of overfitting and the like, let’s consider R^2. This quantity is not calculated by **glm()**, but recall from Section 2.9.2 that R^2 is the squared correlation between the predicted and actual Y values. This quantity makes sense for any regression situation, so let’s calculate it here:

```
> cor(poisout$fitted.values,poisout$y)^2
[1] 0.2314203
> cor(quasiout$fitted.values,quasiout$y)^2
[1] 0.3008466
```

The “unorthodox” model performed better than the “official” one! We cannot generalize from this, but it does show again that one must use transformations carefully.

4.5 Least-Squares Computation

A point made in Section 1.4 was that the regression function, i.e., the conditional mean, is the optimal predictor function, minimizing mean squared prediction error. This still holds in the nonlinear (and even nonparametric) case. The problem is that in the nonlinear setting, the least-squares estimator does not have a nice, closed-form solution like (2.28) for the linear case. Let’s see how we can compute the solution through iterative approximation.

4.5.1 The Gauss-Newton Method

Denote the nonlinear model by

$$E(Y \mid X = t) = g(t, \beta) \tag{4.40}$$