Unifying Theory of GLMs (Part IV)

Rebecca C. Steorts

Computing set up

```
library(tidyverse)
library(tidymodels)
library(knitr)
library(viridis)

knitr::opts_chunk$set(fig.width = 8,
                      fig.asp = 0.618,
                      fig.retina = 3,
                      dpt = 300,
                      out.width = "70%",
                      fig.align = "center")

ggplot2::theme_set(ggplot2::theme_bw(base_size = 16))

colors <- tibble::tibble(green = "#B5BA72")
```

Generalized Linear Regression Model and General Linear Regression Model

Generalized linear regression model (GLM) and general linear regression are usually treated as different regression methods. In fact, the general linear regression is just one special case of GLM. For the general linear regression, we can write the model as:

$$Y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i$$

where ϵ_i follows a normal distribution, typically $N(0, \sigma^2)$.

Generalized Linear Regression Model and General Linear Regression Model

For the generalized linear model (GLM), there is no explicit error term ϵ_i. Instead, we assume that the response variable Y follows a specific distribution (e.g., Poisson, Bernoulli, Gamma). A transformation of the expected value of Y, called the **link function**, is assumed to be linearly related to the predictor variables:

$$g(\mu_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

Estimate Coefficients for a Poisson Regression

To estimate β and its confidence interval, the **Maximum Likelihood Estimation (MLE)** method is commonly used. Since an explicit solution is often unavailable, we use the **Newton-Raphson** algorithm, leading to the **Iterative Re-weighted Least Squares (IRLS)** method.

Assuming the response variable Y follows a Poisson distribution:

$$P(Y_i = y_i) = \frac{e^{-\mu_i}\mu_i^{y_i}}{y_i!}$$

Estimate Coefficients for a Poisson Regression

The likelihood function is:

$$L = \prod_{i=1}^{n} \frac{e^{-\mu_i} \mu_i^{y_i}}{y_i!}$$

Taking the logarithm:

$$\ell = \sum_i \left(-\mu_i + y_i \log \mu_i - \log y_i! \right)$$

Estimate Coefficients for a Poisson Regression

Let ${x_i}^T$ be the row vector $(x_{io}, x_{i1}, \ldots, x_{ip})$ and β is the column vector $(\beta_o, \beta_1, \ldots, \beta_p)$.

Using the **log link function**:

$$g(\mu_i) = \log(\mu_i) = {x_i}^T\beta \implies \mu_i = e^{{x_i}^T\beta}$$

Taking derivatives with respect to β_j $j = 1, \ldots, p$ we find

$$\begin{aligned}
\frac{\partial \ell}{\partial \beta_j} &= -\sum_i e^{{x_i}^T\beta} x_{ij} + \sum_i y_i x_{ij} && (1)\\
&= \sum_i x_{ij}(y_i - e^{{x_i}^T\beta}) && (2)
\end{aligned}$$

Estimate Coefficients for a Poisson Regression

We can write the score functions alterantively in matrix notation as follows

$$\nabla \ell_{\beta} = X^T(y - e^{X\beta})$$

The Hessian can be written as

$$\nabla \ell_{\beta^2} = X^T(y - e^{X\beta}) = 0 - X^T e^{X\beta} X = -X^T W X$$

Let $W = e^{X\beta}$, which is an $n \times n$ matrix with $e^{x_i^T \beta}$ on each diagonal.

Newton Raphson

Newton Raphson becomes as follows

$$\beta^{t+1} = \beta^{(t)} + (X^T W_{(t)} X)^{-1} [X^T (y - e^{X\beta^{(t)}})],$$

where $W_{(t)} = e^{X\beta^{(t)}}$

Iterative re-weighted least squares(IRLS) algorithm

- Make a transformation to show that NR becomes IRLS. Can go back and forth.
- Default in R is IRLS.
- Get some basic code going and come it to glm()

Example

```r
y <- c(2,3,6,7,8,9,10,12,15)
x <- matrix(c(1,1,1,1,1,1,1,1,1, -1, -1, 0, 0, 0, 0, 1, 1, 1), nrow=9, ncol=2)
data_4.3 <- data.frame(y, x1 = c(-1, -1, 0, 0, 0, 0, 1, 1, 1))

poisson_Newton_Raphson<-function(x,y,b.init,tol=1e-8){
  change <- Inf
  b.old <- b.init
  while(change > tol) {
    eta <- x %*% b.old  # linear predictor
    w<-diag(as.vector(exp(eta)),nrow=nrow(x),ncol=nrow(x))
    b.new<-b.old+solve(t(x)%*%w%*%x)%*%t(x)%*%(y-exp(eta))
    change <- sqrt(sum((b.new - b.old)^2))
    b.old <- b.new
  }
  b.new
}
poisson_Newton_Raphson(x,y,rep(1,2))
```

```
##           [,1]
## [1,] 1.8892720
## [2,] 0.6697856
```

Example

```r
poisson_IRLS <- function(x, y, b.init, tol=1e-8){
  change <- Inf
  b.old <- b.init
  while(change > tol) {
    eta <- x %*% b.old
    w <- diag(as.vector(exp(eta)), nrow=nrow(x), ncol=nrow(x))
    z <- solve(w) %*% (y - exp(eta))
    weight <- exp(eta)
    b.new <- b.old + lm(z ~ x - 1, weights = weight)$coef
    change <- sqrt(sum((b.new - b.old)^2))
    b.old <- b.new
  }
  b.new
}

poisson_IRLS(x, y, rep(1,2))
```

```
##        x1        x2
## 1.8892720 0.6697856
```

Example

```r
m1<-glm(y ~ x1, family=poisson(link="log"), data=data_4.3)
summary(m1)
```

```
##
## Call:
## glm(formula = y ~ x1, family = poisson(link = "log"), data = data_4.3)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.8893     0.1421  13.294  < 2e-16 ***
## x1            0.6698     0.1787   3.748 0.000178 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 18.4206  on 8  degrees of freedom
## Residual deviance:  2.9387  on 7  degrees of freedom
## AIC: 41.052
##
## Number of Fisher Scoring iterations: 4
```