R_HW3_Binomial and Proportion Responses

Eom SangJun

2020 9 26

#1. Binomial Regression Model

Chapter 2 에서는 outcome 이 0 또는 1 인 경우를 다뤘다면 이번에는 Response 가 Bernoulli distributed 된 것이 아니라 Binomial 인 경우를 다룰 것이다. 이 때 Resoponse Y_i 의 확률은 다음과 같이 표시할 수 있다.

$$P(Y_i = y_i) = \binom{m_i}{y_i} p_i^{y_i}(1-p_i)^{m_i - y_i}$$

이 때 Y_i for $i = 1 \dots n$ 은 Response Variable 은 binomially distributed $B(m_i, p_i)$

각 Response Variable 들은 독립적이라고 가정한다.

Response Variable 을 구성하는 각각의 개별의 outcomes 또는 시도들(trials)은 모두 같은 q predictors($x_{i1}, \dots, x_{iq}$) 에 종속되는데, 이 때 trials 의 그룹을 covariate class 라고 부른다.

Binary case 에서 linear predictor 를 만들면 다음과 같다.

$$\eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_q x_{iq}$$

그리고 logistic link function 을 사용한다고 했을 때 log-likelihood 는 다음과 같다.

$$l(\beta) = \sum_{i=1}^{n} [y_i \eta_i - m_i \log(1 + e_i^{\eta}) + \log \binom{m_i}{y_i}]$$

이제 예시를 통해 살펴보자.

이번에 사용할 데이터는 온도와 orings 에 관한 것이다. 온도가 일정 수준 이상으로 낮아지면 orings 는 손상될 위험이 있고, 특정 온도에서 총 6 개 중에 몇 개의 orings 가 손상되었는지를 나타내는 데이터를 사용할 것이다.

```
data(orings, package = 'faraway')
```

그리고 특정 온도에서 손상된 orings 의 비율을 나타내는 그래프를 그려보자.

```
plot(damage/6 ~ temp, orings, xlim=c(25,85), ylim=c(0,1), xlab='Temperature',
     ylab='Prob of damage')
```

우리는 특정 온도에서 oring 이 몇 개나 손상될 것인지를 예측하고자 하므로 Binomial Regression 을 이용해서 예측하도록 하자. 이 때 glm function 에서 Response 에 넣어주어야 하는 것은 두 가지이다. 하나는 success 의 개수, 하나는 failure 의 개수이다. 따라서 두 개의 정보를 n*2 짜리 matrix 로 만들어서 Response 자리에 넣어주어야 한다. 이렇게 했을 때 glm command 를 이용해서 Binomial Regression 을 적용하면 다음과 같다.

```
lmod <- glm(cbind(damage, 6-damage) ~ temp, family=binomial, orings)
```

결과는 다음과 같다.

```
sumary(lmod)
```

```
##              Estimate Std. Error z value  Pr(>|z|)
## (Intercept) 11.662990   3.296263  3.5382 0.0004028
## temp        -0.216234   0.053177 -4.0663 4.777e-05
##
## n = 23 p = 2
## Deviance = 16.91228 Null Deviance = 38.89766 (Difference = 21.98538)
```

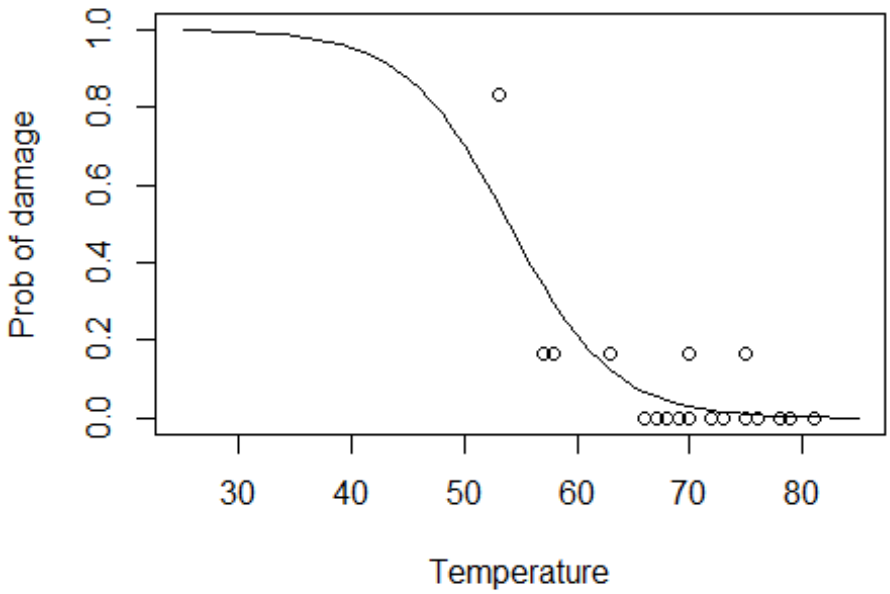
온도의 영향이 유의하다고 나온다.

Binomial Regression 을 ilogit command 를 이용하여 그래프를 그리면 다음과 같다.

```
x <- seq(25,85,1)
```

여기서 x 는 온도로, 화씨 25 도에서 85 도 사이로 충분히 넓게 잡는다.

```
lines(x, ilogit(11.6630-0.2162*x))
```



➔ Point 들은 온도에 따른 손상된 orings 의 비율을 나타낸다. 높은 온도에서 0 에 가까운 점들이 많다는 것을 알 수 있다.

➔ Binomial Regression 을 이용해서 line 을 그렸을 때 40~60 사이에서 확률이 급격히 낮아지는 것을 볼 수 있다.

31 도에서의 확률을 구해보면 다음과 같다.

```
ilogit(11.6630-0.2162*31)
```

```
## [1] 0.9930414
```

매우 높다는 것을 알 수 있다.

#2. Inference

Chapter 2 에서 우리는 binomial deviance 를 구하기 위해서 likelihood 를 이용하였는데, 이번에도 마찬가지이다.

$$D = 2\sum_{i=1}^{n}\{y_i log y_i/\hat{y_i} + (m_i - y_i)\log(m_i - y_i)/(m_i - \hat{y_i})\}$$

이 때 $\hat{y}_i$는 fitted value from the model

만약 model 이 맞는다면, Y 가 정말로 binomial 이고 m_i 가 비교적 크다면 deviance 는 근사적으로 카이제곱 분포를 따른다. 이 때 자유도는 n-q-1

따라서 우리는 deviance 를 model 이 적절한 fit 을 가졌는 지를 test 하는 데에 사용할 수 있다.

p-value 를 구해서 test 를 진행해보자.

```
pchisq(deviance(lmod), df.residual(lmod), lower=FALSE)
```

```
## [1] 0.7164099
```

0.05 를 넘기 때문에 current model 에 대한 가설을 채택한다. 즉, 현재 모델이 충분히 data 에 fit 한다고 판단한다. 물론 이는 우리의 모델이 정확히 정답이거나 더 간단한 모델이 틀리다는 것을 말해주지는 않는다.

만약 Null Model 일 때는 어떨까? 앞선 sumary 에서 Null Deviance 값은 약 38.9 였고 Null Model 에서 q=0 이므로 n-q-1=23-0-1=22. 똑같이 카이제곱 검정을 해보면,

```
pchisq(38.9, 22, lower=FALSE)
```

```
## [1] 0.01448877
```

보다시피 0.05 미만이다. 따라서 적절하게 fit 하지 않는다. 즉, Response 가 어떠한 predictor 에도 의존하지 않는 simple variation 이라고는 볼 수 없다.

d 의 자유도를 갖는 카이제곱 분포의 평균은 d 이고 표준편차는 $\sqrt{2d}$라는 것을 생각하면 p-value 를 계산하지 않아도 deviance 가 큰지 작은지를 대략적으로 판단할 수 있다.

만약 deviance 가 자유도보다 훨씬 크다면, 귀무가설은 기각될 수 있다.

카이제곱 분포 근사는 말그대로 근사이기 때문에 m_i 가 커질수록 정확해지고 작아질수록 부정확해진다. 따라서 m_i=1 인 경우는 완전히 실패하게 된다.

비록 m_i 가 어느정도 커야 되는 지에 대한 것은 없지만, 대략적으로 모든 i 에 대해 5 가 넘어가는 것이 종종 제시된다. Permutation 이나 bootstrap 방법이 대안으로 제시될 수도 있다.

deviance 는 두 모델을 비교하는 데에도 사용될 수 있다. 방식은 Chapter2 에서 나왔던 방식과 동일하다.

Null deviance 에서 current model deviance 를 뺀 것과 Null model 의 자유도에서 current 모델의 자유도를 뺀 것을 이용하여 카이제곱 검정을 하면,

```
pchisq(38.9-16.9,1,lower=FALSE)
```

```
## [1] 2.726505e-06
```

0.05 보다 작기 때문에 온도 predictor 는 유의하다는 것을 알 수 있다.

만약 어떤 covariate class 에서 나온 모든 cases 들을 group 을 안 지으면 어떻게 될까?

```
erings <- with(orings,
               data.frame(temp=rep(temp, each=6),
                          damage=as.vector(sapply(orings$damage, function(x)
rep(c(0,1), times=c(6-x,x))))))
head(erings)
```

```
##   temp damage
## 1   53      0
## 2   53      1
## 3   53      1
## 4   53      1
## 5   53      1
## 6   53      1
```

결과에서 볼 수 있듯이, 특정 온도에서 각각의 시행에 대한 결과가 합쳐지는 것이 아니라 따로 나온다는 것을 알 수 있다.

```
emod <- glm(damage ~ temp, family=binomial, erings)
sumary(emod)
```

```
##               Estimate Std. Error z value  Pr(>|z|)
## (Intercept) 11.662990   3.296157  3.5384 0.0004026
## temp        -0.216234   0.053175 -4.0665 4.773e-05
##
## n = 138 p = 2
## Deviance = 54.75942 Null Deviance = 76.74480 (Difference = 21.98538)
```

이를 통해 binomial regression 을 하고 결과를 확인해보면, parameter estimates, standard errors, deviance difference 는 같게 나와서 결론은 똑같게 할 수 있다. 그러나 deviance 의 절대값이 다르고 n 의 개수도 차이가 난다. 우리는 이 버전으로는 residual deviance 를 이용한 goodness of fit test 를 할 수 없다.

각 coefficient 의 신뢰구간을 구하면 다음과 같다.

```
confint(lmod)
```

```
## Waiting for profiling to be done...
```

```
##                 2.5 %    97.5 %
## (Intercept)  5.575195 18.737598
## temp        -0.332657 -0.120179
```

#3. Pearson's Chi-square Statistic

deviance 말고도 model fit 을 측정할 수 있는 대안으로는 Pearson's Chi-square statistic 이 있다.

Pearson's Chi-square statistic 의 일반적인 form 은 다음과 같다.

$$X^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

이 때 O_i 는 관측된 수이고 E_i 는 기대값이다.

Binomial case 에 적용하면, $O_i = y_i, E_i = n_i\hat{p}_i$ for successes, $O_i = n_i - y_i, E_i = n_i(1-\hat{p}_i)$ for failures 이다. 따라서

$$X^2 = \sum_{i=1}^{n} \frac{(y_i - n_i\hat{p}_i)^2}{n_i\hat{p}_i(1-\hat{p}_i)}$$

만약 우리가 Pearson residual 을 다음과 같이 정의한다면,

$$r_i^p = (y_i - n_i\hat{p}_i)/\sqrt{var\widehat{y_i}}$$

꼴이며, 이는 일종의 standard residual 이라고 볼 수 있다. 그러면 $X^2 = \sum_{i=1}^{n}(r_i^p)^2$이다. 따라서 Pearson's chi-square statistic 은 normal linear model 에서의 residual sum of squares 와 대응된다고 볼 수 있다. 우리는 deviance 때와 마찬가지로, 똑같은 null distribution 에서 deviance 대신에 X^2을 사용하여 test 를 진행할 수도 있다. 다만, 이 때 주의해야 하는 것은 우리의 모델이 X^2이 아닌 deviance 를 최소화하는 것에 맞춰진 model 이라는 것이다. 즉, X^2은 predictor 가 추가될수록 증가할 수 있다.

우리의 모델에서 X^2의 값은 28.067 이다. 이를 deviance 와 비교해보자.

```
deviance(lmod)
```

```
## [1] 16.91228
```

28.067 과는 좀 차이가 있는 것을 알 수 있다.

그렇다면 X^2을 이용해서 test 를 진행한 결과는 어떨까?

```
1-pchisq(28.067, 21)
```

```
## [1] 0.1382613
```

다행히 p-value 가 0.05 이상이므로 deviance 를 이용하여 test 한 결과와 같은 결론을 내릴 수 있다.

#4. Overdispersion

우리는 가끔, 모델이 맞다고 가정했을 때보다 더 큰 deviance 를 관찰할 때가 있다. 이 때 우리는 모델 가정에서 무엇이 잘못되었는 지를 확인해볼 필요가 있다.

우선 가장 일반적인 설명은 model 의 구조적인 형태(structural form)이 잘못되었다는 것이다. 적절하지 않은 predictor 를 넣었을 수도 있고, predictor 들을 적당한 형태로 변형 또는 결합하지 않았을 수 있다.

또 다른 일반적인 설명은, 적은 수의 outlier 들의 존재이다. 이는 diagnostic methods 를 이용해서 간단하게 확인할 수 있다. 만약 더 많은 수의 outlier 들이 발견된다면, 그들은 예외적인 것이 아니게 되며, 따라서 error distribution 에 잘못된 것이 있다고 결론지을 수 있다.

Group size 가 너무 작은 경우에도 큰 deviance 를 초래할 수 있다.

이 모든 경우들을 제외하고 나서 또 다른 경우의 수는 model 의 random part 에 결함이 있다는 것이다.

Binomial case 에서 var Y = mp(1-p)여야 한다. 이 때 m 은 group 의 size. 그런데 종종 직접 계산한 variance 값이 mp(1-p)값보다 훨씬 클 때가 있다. 이를 overdispersion 이라고 부른다.

Overdispersion 이 일어나는 주요한 이유는 두 가지이다. 우선 우리는 그룹 안에서 success 또는 failure 가 발생하는 case 또는 probability 가 independent 하고 identical 하다고 가정한다. 하지만 이 가정이 틀린 경우 일어날 수 있다. 우선 constant p assumption 을 살펴보자. 우리는 각 그룹에서 p 값이 동일하다고 생각한다. 그러나 그룹별로 설명되지 않은 이질성이 있을 수 있고, 이는 p 의 변동을 불러올 수 있다. 또한 이러한 이질성은 clustering 에서 발생될 수 있다.

Sample size 를 m, cluster size 를 k, 그리고 cluster 의 개수를 l=m/k 라고 하자. 그러면 number of success in cluster i 를 $Z_i \sim B(k, p_i)$라고 정의할 수 있다. 이제 p_i가 constant 가 아니라 random variable 이라고 해보자. $E(p_i) = p,\ \ Var(p_i) = \tau^2 p(1-p)$ 라고 하자. 그러면 total number of success = Y = $Z_1 + Z_2 + \cdots Z_l$이라고 할 때

$$E(Y) = \sum E(Z_i) = \sum_{i=1}^{l} kp = mp$$

즉, 평균은 standard case 와 동일하다.

그러나, 분산은 조금 다르다.

$$Var(Y) = \sum var(Z_i) = \sum\{E\big(var(Z_i|p_i)\big) + var\big(E(Z_i|p_i)\big)\} = (1 + (k-1)\tau^2)mp(1-p)$$

$(1 + (k-1)\tau^2) \geq 1$이기 때문에 Y 는 overdispersed 되었다.

Overdispersion 은 trial 간의 dependency 때문에 일어날 수도 있다. 만약 response 가 동일한 cause 를 가진다면, response 들은 positively correlated 될 수 있다.

Overdispersion 을 modeling 하는 가장 간단한 방법은 추가적인 dispersion parameter(σ^2)를 도입하는 것이다. 즉, $Var(Y) = \sigma^2 mp(1-p)$ Standard case 에서는 $\sigma^2 = 1$

σ^2는 Pearson Chi-square statistic 을 이용해서 추정할 수 있다.

$$\hat{\sigma}^2 = \frac{X^2}{n-p}$$

이 때 X^2대신 deviance 를 쓰는 것은 추천하지 않는다. 왜냐하면 consistent 하지 않을 수 있기 때문. Beta 의 추정은 σ^2가 response mean 을 바꾸지는 않기 때문에 변화 없다. 그러나 분산의 추정은 다르다.

$$\widehat{Var(\hat{\beta})} = \hat{\sigma}^2 \left(X^T \hat{W} X\right)^{-1}$$

따라서 우리는 standard error 를 $\hat{\sigma}$의 요소만큼 늘려주어야 한다.

모델 비교에서도 deviance 의 차이를 그대로 사용할 수는 없다. 왜냐하면 test statistic 이 $\sigma^2\chi^2$분포를 따르기 때문이다. 대신 F statistic 을 사용해야 한다.

$$F = \frac{(D_{small} - D_{large})/(df_{small} - df_{large})}{\hat{\sigma}^2}$$

이 통계량은 근사적으로 F 분포를 따른다.

이제 예시를 통해 Overdispersion case 를 살펴보자.

Troutegg 데이터는 지역과 period 별로 송어 알의 생존 정도를 나타내주는 데이터이다. Total 은 묻힌 송어알의 총 개수이고 survive 는 그 중 몇 개가 살아남았는 지를 의미한다.

Data 의 형태는 다음과 같다.

```
data(troutegg, package='faraway')
ftable(xtabs(cbind(survive, total) ~ location+period, troutegg))
```

```
##                  survive total
## location period
## 1        4            89    94
##          7            94    98
##          8            77    86
##          11          141   155
## 2        4           106   108
##          7            91   106
##          8            87    96
##          11          104   122
```

```
## 3        4           119   123
##          7           100   130
##          8            88   119
##          11           91   125
## 4        4           104   104
##          7            80    97
##          8            67    99
##          11          111   132
## 5        4            49    93
##          7            11   113
##          8            18    88
##          11            0   138
```

이를 이용하여 binomial regression 을 진행하면 다음과 같다.

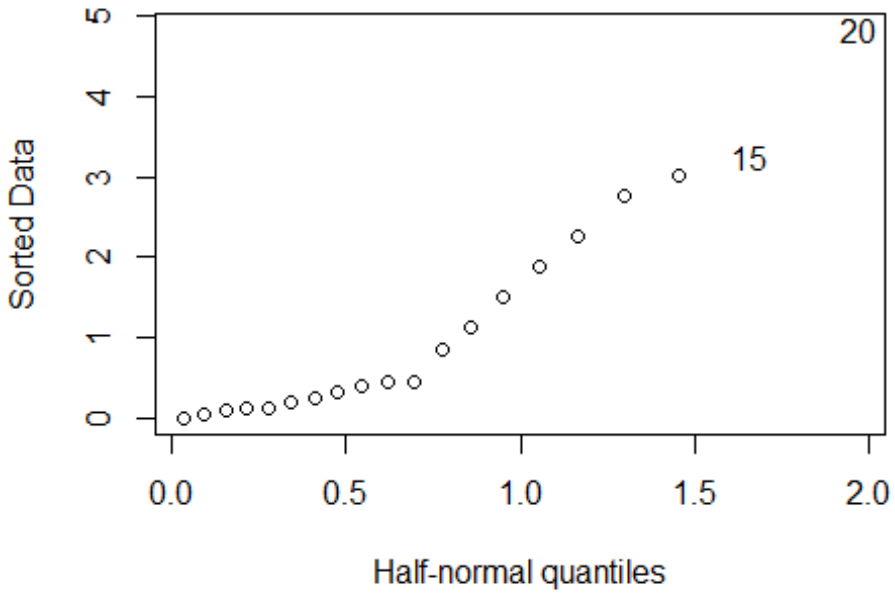
```
bmod <- glm(cbind(survive, total-survive) ~ location+period, family=binomial,
 troutegg)
sumary(bmod)
```

```
##             Estimate Std. Error  z value  Pr(>|z|)
## (Intercept)  4.63582    0.28132  16.4790 < 2.2e-16
## location2   -0.41678    0.24610  -1.6936   0.09035
## location3   -1.24208    0.21944  -5.6603 1.511e-08
## location4   -0.95086    0.22876  -4.1566 3.230e-05
## location5   -4.61381    0.25021 -18.4394 < 2.2e-16
## period7     -2.17018    0.23840  -9.1031 < 2.2e-16
## period8     -2.32562    0.24295  -9.5726 < 2.2e-16
## period11    -2.44995    0.23410 -10.4656 < 2.2e-16
##
## n = 20 p = 8
## Deviance = 64.49512 Null Deviance = 1021.46868 (Difference = 956.97356)
```

Deviance 가 64.5 on 12 degrees of freedom 인 것을 봤을 때, model 이 fit 하지 않다는 것을 알 수 있다. Overdispersion 임을 결론짓기 전에 다른 가능성들을 살펴보자.

우선 Outlier 를 살펴보자. Outlier 는 halfnorm command 를 이용하여 살펴볼 수 있다.

```
halfnorm(residuals(bmod))
```



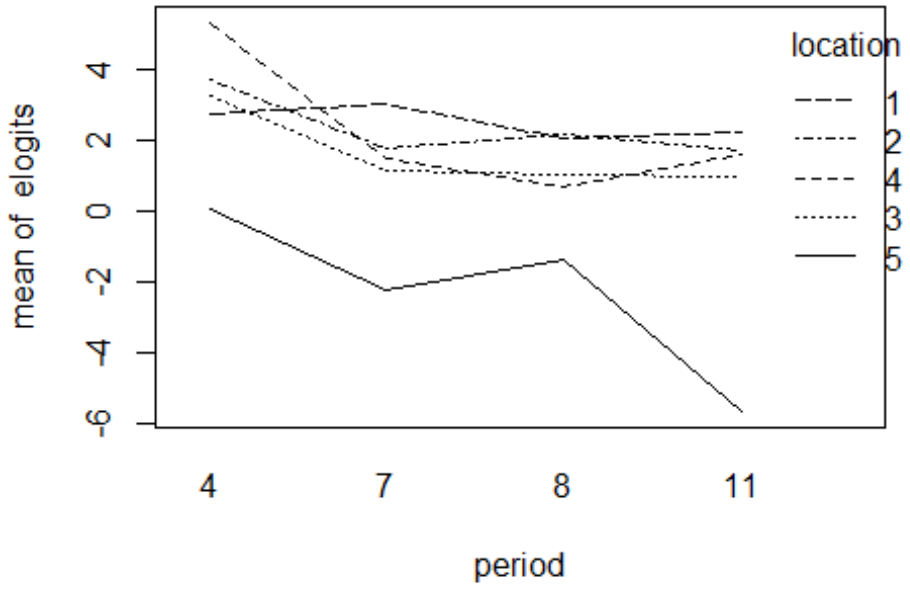
➔ 명확한 single outlier 는 없는 것 같다.

우리는 또한 predictor 들이 제대로 표현되었는 지를 empirical logits 을 그려보고 확인할 수 있다.

Empirical logits = $\log\left(\frac{y+1/2}{m-y+1/2}\right)$

이 때 1/2 은 group 이 모두 성공이거나 실패여서 infinite 값이 생기는 것을 방지하기 위해서 넣어준다. 이제 empirical logits 의 interaction plot 을 그려보면 다음과 같다.

```
elogits <- with(troutegg, log((survive+0.5)/(total-survive+0.5)))
with(troutegg, interaction.plot(period, location, elogits))
```



➔ 확실하게 뚜렷한 interaction 은 보이지 않는다.

➔ 따라서 linear model 을 설정한 것이 부적절한 것 같지 않다.

다른 가능성들을 배재했으니 이제 overdispersion 에 대해 고려해보자. Overdispersion 이 일어날 수 있는 가능성들은 다양하다. 송어 알들의 이질성이 있을 수 있고, 실험 과정에서의 variation 이 있을 수 있다는 점도 overdispersion 을 야기할 수 있다.

Overdispersion parameter 를 추정하면 다음과 같다.

```
(sigma2 <- sum(residuals(bmod, type='pearson')^2)/12)
```

```
## [1] 5.330322
```

이는 standard binomial GLM 에서의 것보다 상당히 크다는 것을 알 수 있다.

이제 predictor 에 대한 F-test 를 해보자.

Overdispersion 의 경우 scale argument 를 추가해주어야 한다.

```
drop1(bmod, scale=sigma2, test='F')
```

```
## Warning in drop1.glm(bmod, scale = sigma2, test = "F"): F test assumes
## 'quasibinomial' family
```

```
## Single term deletions
##
## Model:
## cbind(survive, total - survive) ~ location + period
##
## scale:  5.330322
##
##          Df Deviance    AIC F value    Pr(>F)
## <none>         64.50 157.03
## location  4   913.56 308.32  39.494 8.142e-07 ***
## period    3   228.57 181.81  10.176  0.001288 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Location 과 period 모두 유의하다는 것을 알 수 있다.

또한 warning message 에서 볼 수 있듯이, quasi-binomial GLM 을 사용하였다.

Free dispersion parameter 를 가졌기 때문에 이제 goodness of fit test 는 불가능하다.

우리는 이제 dispersion parameter 를 이용하여 standard error 추정치를 증가시킬 수 있다.

```
summary(bmod, dispersion=sigma2)
```

```
##             Estimate Std. Error z value  Pr(>|z|)
## (Intercept)  4.63582    0.64949  7.1376 9.494e-13
## location2   -0.41678    0.56817 -0.7335   0.46323
## location3   -1.24208    0.50663 -2.4517   0.01422
## location4   -0.95086    0.52814 -1.8004   0.07180
## location5   -4.61381    0.57768 -7.9868 1.385e-15
## period7     -2.17018    0.55040 -3.9429 8.051e-05
## period8     -2.32562    0.56090 -4.1462 3.380e-05
## period11    -2.44995    0.54047 -4.5330 5.815e-06
##
## Dispersion parameter = 5.33032
## n = 20 p = 8
## Deviance = 64.49512 Null Deviance = 1021.46868 (Difference = 956.97356)
```

이전 결과와는 다르게 오직 5 번째 location 만 유의한 것으로 나타났다.

이 dispersion parameter method 는 covariate class 들의 사이즈가 거의 동일할 때만 적절하게 사용 가능하다. 만약 그렇지 않은 경우 더 정교한 방법이 필요한데, R 에서는 dispmod package 안에 있는 glm.binomial.disp command 를 사용하여 적용할 수 있다.

```
library(dispmod)
```

```
## Warning: package 'dispmod' was built under R version 3.6.3
```

```
dmod <- glm.binomial.disp(bmod)
```

```
##
## Binomial overdispersed logit model fitting...
## Iter.  1  phi: 0.03983754
## Iter.  2  phi: 0.03813596
## Iter.  3  phi: 0.03814806
## Iter.  4  phi: 0.03814797
## Iter.  5  phi: 0.03814797
## Converged after 5 iterations.
## Estimated dispersion parameter: 0.03814797
##
## Call:
## glm(formula = cbind(survive, total - survive) ~ location + period,
##     family = binomial, data = troutegg, weights = disp.weights)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.04625  -0.21394   0.01708   0.28386   1.36990
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   4.5183     0.6206   7.281 3.32e-13 ***
## location2    -0.3769     0.5603  -0.673   0.5011
## location3    -1.2099     0.5066  -2.388   0.0169 *
## location4    -0.9562     0.5199  -1.839   0.0659 .
## location5    -4.4679     0.5586  -7.999 1.25e-15 ***
## period7      -2.0858     0.5201  -4.011 6.06e-05 ***
## period8      -2.2273     0.5225  -4.263 2.02e-05 ***
## period11     -2.3623     0.5186  -4.555 5.23e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 190.19  on 19  degrees of freedom
## Residual deviance:  12.40  on 12  degrees of freedom
## AIC: 43.393
##
## Number of Fisher Scoring iterations: 5

sumary(dmod)

##             Estimate Std. Error z value  Pr(>|z|)
## (Intercept)  4.51827    0.62059  7.2806 3.322e-13
## location2   -0.37693    0.56029 -0.6727   0.50111
## location3   -1.20993    0.50657 -2.3885   0.01692
## location4   -0.95622    0.51993 -1.8392   0.06589
## location5   -4.46793    0.55856 -7.9990 1.254e-15
## period7     -2.08576    0.52006 -4.0106 6.055e-05
## period8     -2.22727    0.52251 -4.2626 2.020e-05
```

```
## period11    -2.36234    0.51857 -4.5555 5.226e-06
##
## n = 20 p = 8
## Deviance = 12.40002 Null Deviance = 190.18576 (Difference = 177.78574)
```

Disp 를 사용하지 않았을 때와 큰 차이가 없는 것을 알 수 있는데, 이는 covariate class 의 size 가 거의 동일하기 때문이다.

#5. Quasi-Binomial

Quasi-binomial model 은 extra-binomial variation 을 가능하게 해주는 방법 중 하나이다.

기본적인 아이디어는 response 의 mean 과 variance 가 어떻게 linear predictor 와 연결되어있는지를 구체화하는 것이다.

일반적인 binomial model 에서는 binomial distribution 에서 추가적인 정보 말고 오직 mean 과 variance information 만을 활용한다. 따라서 beta 와 standard error 를 추정할 때 full binomial assumption 은 필요하지 않다.

다만 문제는 우리가 추론을 할 때 일어난다. 신뢰구간을 찾거나 또는 가설 검정을 할 때, 우리는 몇 개의 분포적 가정이 필요하다. 이전에 우리는 deviance 를 사용했지만 이를 위해서 우리는 likelihood 가 필요하며 또 likelihood 를 계산하기 위해서 distribution 이 필요하다. 이제 우리는 분포를 가정하지 않고도 계산될 수 있는 likelihood 의 적절한 대체재가 필요하다.

Y_i의 평균이 μ_i이고 분산이 $\phi V(\mu_i)$ 라고 하자. 각 Y_i는 독립적이라고 가정한다. 이 때 우리는 score U_i를 다음과 같이 정의하자.

$$U_i = \frac{Y_i - \mu_i}{\phi V(\mu_i)}$$

그러면

$$E(U_i) = 0$$

$$Var(U_i) = \frac{1}{\phi V(\mu_i)}$$

$$-E\left(\frac{\partial U_i}{\partial \mu_i}\right) = -E\left(\frac{-\phi V(\mu_i) - (Y_i - \mu_i)\phi V'(\mu_i)}{[\phi V(\mu_i)]^2}\right) = \frac{1}{\phi V(\mu_i)}$$

이러한 특징들은 log-likelihood 의 미분(l')에 의해 공유된다. 이는 우리가 U 를 l' 대신에 사용하는 것을 제시한다. 따라서 우리는

$$Q_i = \int_{y_i}^{\mu_i} \frac{y_i - t}{\phi V(t)} dt$$

라고 정의할 수 있다.

목적은 Q 가 log-likelihood 처럼 행동해야 한다는 것이다. 우리는 log quasi-likelihood for all n observations 를 다음과 같이 정의할 수 있다.

$$Q = \sum_{i=1}^{n} Q_i$$

maximum likelihood 에서 기대되는 일반적인 asymptotic 성질은 여기에도 똑같이 적용될 수 있다.

Quasi-likelihood 는 오직 variance function 에 직접적으로 의존하고 분포의 선택이 오직 variance function 을 결정한다는 것에 유의하자. 따라서 variance function 의 선택은 model 에서 random structure 와 연관되고 link function 은 model 의 systematic part 와의 관계를 결정한다.

Standard linear model 에서 quasi-likelihood 는 정확히 log-likelihood 와 대응된다. 여기서 dispersion parameter ϕ는 σ^2이다. 따라서 이 접근으로는 아무것도 얻을 것이 없다.

그러나 binomial model 에서는 ϕ도입은 model 에 추가적인 dimension of flexibility 를 제공한다. 이는 overdispersion 을 modeling 하는 데에 도움이 된다. 한 가지 흥미로운 가능성은 어떤 $V(\mu)$의 선택들은 어떠한 알려져 있는 분포와도 일치하지 않을 수 있다는 점이다.

베타의 추정치는 Q 를 최대화함으로써 얻어진다. 모든 절차는 overdispersion parameter 를 넣지 않았을 때와 동일하게 진행되지만, ϕ추정은 다르다. 왜냐하면 likelihood approach 는 여기서는 믿음직하지 않기 때문이다. 대신

$$\hat{\phi} = \frac{X^2}{n-p}$$

를 추천한다.

비록 quasi-likelihood estimator 가 더 적은 가정을 요하기 때문에 매력적일 수 있지만 regular likelihood based estimator 에 대응하는 것에 비해서는 일반적으로 덜 efficient 하다(분산이 큼). 따라서 만약 분포에 대한 정보를 가지고 있다면 그것을 쓰는 것이 낫다.

Regular deviance for a model 은 현재 model 의 log-likelihood 와 saturated model 의 log-likelihood 의 차이로 형성된다.

$$D(y, \hat{\mu}) = -2\phi \sum_i (l(\hat{\mu_i}|y_i) - l(y_i|y_i))$$

그리고 유사성에 의해 quasi-deviance 는 $-2\phi Q$이다. 왜냐하면 saturated model 로부터의 기여도가 0 이기 때문이다. ϕ가 지워지고, quasi-deviance 는 다음과 같이 표현될 수 있다.

$$Q = -2\sum_i \int_{y_i}^{\mu_i} \frac{y_i - t}{V(t)} dt$$

이제 data 를 이용하여 위의 내용을 직접 확인해보자.

우선 우리가 이용하려는 데이터는 포유동물의 잠과 관련한 데이터이다. 우리는 포유동물이 잠에 들 때 꿈을 꾸는 시간의 비율을 종속 변수로 두고 체중, 뇌의 무게 등을 predictor 로 하여 살펴보고자 한다.

```
data(mammalsleep, package='faraway')
mammalsleep$pdr <- with(mammalsleep, dream/sleep)
summary(mammalsleep$pdr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.0000  0.1180  0.1755  0.1865  0.2427  0.4615      14
```

Summary 를 통해 살펴보았을 때 꿈꾸는 시간의 비율은 0 에서부터 거의 절반까지 나타난다는 것을 알 수 있다. 어떤 데이터셋에서는 비율의 범위가 0 또는 1 에 절대 가까워지지 않는 것이 있는데, 그러한 데이터셋은 normal gaussian model 을 사용하는 것이 더욱 적절하다. 그런데 이 경우에는 그렇지 않고 매우 적은 비율들을 response value 로 갖고 있으니, Gaussian model 은 적절하지 않다. 따라서 우리는 비율 response 를 그대로 modeling 할 것이다. Logit link 를 사용하는 것은 response 가 0 과 1 사이의 값이기 때문에 합리적으로 보인다. 더 나아가 우리는 측정의 본질적 성질에 의해, 분산이 proportion μ가 적당할 경우 더 크고, 만약 μ가 0 또는 1 에 가까워진다면 더 작아질 것이라고 예상할 수 있다. 이는 variance function 의 근사적인 형태가 $\mu(1-\mu)$임을 제시한다. 이 함수는 0 또는 1 에 가까워지면 작아지고 1/2 에 가까워질수록 커진다. 이런 것들을 봤을 때 canonical logit link 를 가진 binomial GLM 과 대응하지만 response 가 binomial 은 아니다. 따라서 우리는 quasi-binomial 을 사용한다.

우선 모델에 직접적으로 사용하기에 앞서 skewed 된 predictor 들은 log function 을 이용해서 해결해주었다.

```
mod1 <- glm(pdr ~ log(body) + log(brain) + log(lifespan) +
              log(gestation) + predation + exposure + danger,
            family=quasibinomial, mammalsleep)
```

이제 우리는 free dispersion parameter 를 가졌기 때문에 모델 비교에 있어서 F test 를 사용해야 한다.

```
drop1(mod1, test='F')
```

```
## Single term deletions
##
```

```
## Model:
## pdr ~ log(body) + log(brain) + log(lifespan) + log(gestation) +
##     predation + exposure + danger
##                Df Deviance F value  Pr(>F)
## <none>              1.5703
## log(body)       1   1.7786  4.5107 0.04104 *
## log(brain)      1   1.5856  0.3320 0.56827
## log(lifespan)   1   1.6532  1.7949 0.18922
## log(gestation)  1   1.6232  1.1466 0.29181
## predation       1   1.5751  0.1037 0.74946
## exposure        1   1.5851  0.3202 0.57523
## danger          1   1.5848  0.3146 0.57857
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Predation 의 p-value 값이 제일 높으므로(least significant) 이를 제한다. 비슷한 방법으로 backward elimination 을 진행하면 log(brain)과 log(gestation), exposure 를 제할 수 있다.

제외된 predictor 들을 빼고 모델을 다시 돌려보면,

```
mod1 <- glm(pdr ~ log(body) + log(lifespan) + danger, family=quasibinomial, m
ammalsleep)
sumary(mod1)
```

```
##                Estimate Std. Error t value  Pr(>|t|)
## (Intercept)   -0.493233   0.291261 -1.6934 0.0979582
## log(body)      0.146307   0.038425  3.8076 0.0004611
## log(lifespan) -0.286605   0.107975 -2.6544 0.0112584
## danger        -0.173190   0.059952 -2.8888 0.0061512
##
## Dispersion parameter = 0.04065
## n = 45 p = 4
## Deviance = 1.73211 Null Deviance = 2.50881 (Difference = 0.77671)
```

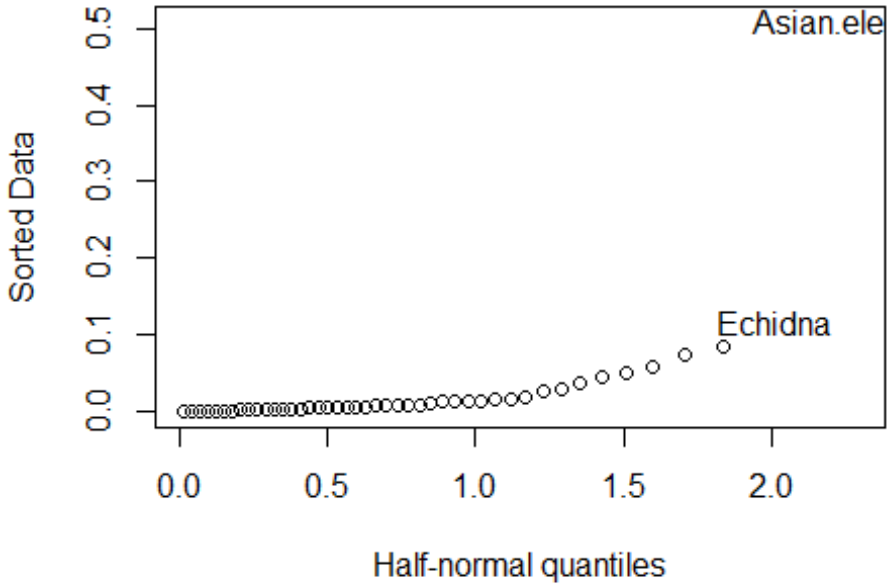
우리가 일반적으로 binomial 에서 보던 기본 값보다 훨씬 더 작은 Dispersion Parameter 값을 확인할 수 있다.

결과를 해석하면, 몸집이 더 크고 수명이 더 짧으며 덜 위험한 환경 속에서 사는 포유동물일수록 꿈을 길게 꾼다.

비교적 큰 residual deviance 값은 이 model 이 잘 fit 하지는 않는다는 것을 알려준다.

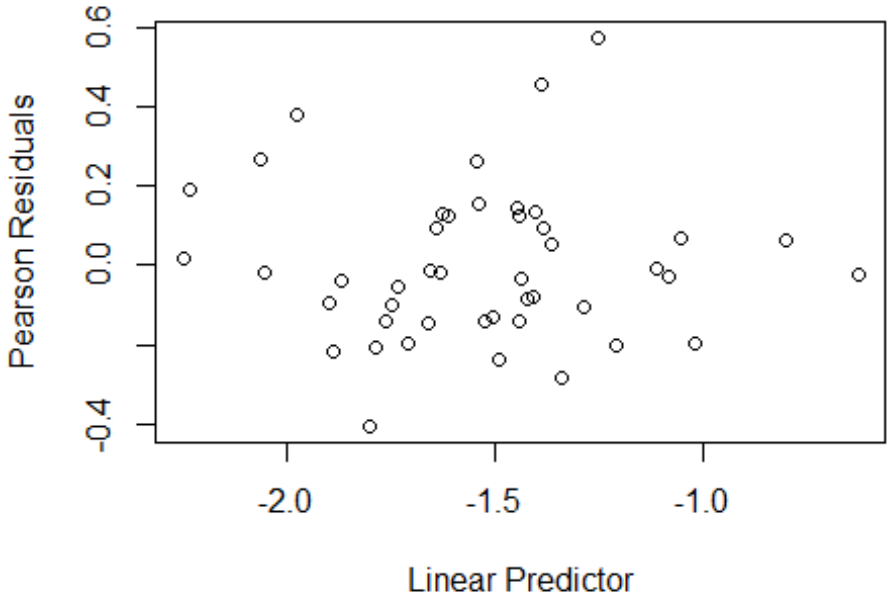
Diagnostic 을 진행해보자.

```
l1 <- row.names(na.omit(mammalsleep[,c(1,6,10,11)]))
halfnorm(cooks.distance(mod1), labs=l1)
```



➔ Asian elephant 는 상당히 influential 하며, 이 case 를 제외한 fit 을 고려해보아야 한다.

```
plot(predict(mod1), residuals(mod1, type='pearson'), xlab='Linear Predictor',
 ylab='Pearson Residuals')
```



➔ Constant variation 형태를 보인다.

➔ 즉, 우리가 선택한 variance function 이 합리적이라는 것을 나타낸다.

➔ Pearson Residual 을 사용한 이유는 이것이 variance function 을 이용해 raw residual 을 더욱 명확히 normalize 하여 더 명확히 check 할 수 있도록 만들어주기 때문이다.

#6. Beta Regression

Response 가 위의 사례처럼 0 과 1 사이의 값을 가질 때, 또는 scaling 을 해서 0 과 1 사이에 둘 수 있을 때 유용한 것이 Beta Regression 이다.

우선 Beta-distributed random variable Y 의 density 는 다음과 같다.

$$f(y|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} y^{a-1}(1-y)^{b-1}$$

이 때 $\mu = \frac{a}{a+b}$, $\phi = a+b$ 라고 하자.

그러면 $E(Y) = \mu$, $Var(Y) = \frac{\mu(1-\mu)}{1+\phi}$ 이다.

그리고 linear predictor 와 평균을 link 할 수 있는데, $\eta = g(\mu)$ 이 때 사용되는 link function 은 binomial regression 에서 사용되는 것이라면 어느 것이라도 적절하다.

R 에서는 mgcv package 를 사용해서 Beta Regression 을 진행할 수 있다.

```
data(mammalsleep, package='faraway')
mammalsleep$pdr <- with(mammalsleep, dream/sleep)
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-28. For overview type 'help("mgcv-package")'.
```

```
modb <- gam(pdr ~ log(body) + log(lifespan), family=betar(), mammalsleep)
```

```
## Warning in family$saturated.ll(y, prior.weights, theta): saturated likelih
ood
## may be inaccurate
```

```
summary(modb)
```

```
##
## Family: Beta regression(8.927)
## Link function: logit
##
## Formula:
## pdr ~ log(body) + log(lifespan)
##
## Parametric coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.37795    0.37322   1.013    0.311
## log(body)      0.26796    0.05513   4.860 1.17e-06 ***
## log(lifespan) -0.92266    0.16585  -5.563 2.65e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =  -0.178   Deviance explained = 73.5%
## -REML = -47.801  Scale est. = 1         n = 45
```

결과를 확인해보면 ϕ값은 Family: Beta Regression 옆에 괄호 안에 있는 값으로 8.927 이다. 그리고 앞의 장에서 quasi 를 이용한 regression 값과 비교해보면 크게 차이가 나지 않는다는 것을 알 수 있다. 다만 Beta-based model 의 장점은 full-distribution model 로, 단순히 point estimate 나 standard error 가 아니라 full predictive distribution 을 만들 수 있다는 것이다.