Homework 4: Data separation and multinomial regression

Solution Set

Due: March 3rd at 11:59 PM

Problem 1: Do the following regarding the Sabermetrics dataset (bball.csv),

(a) Fit the `nnet` model and comment on the similarities and differences between the `nnet` and `VGAM` fits in the Sabermetrics example in the ordinal and multinomial regression notes. Report interesting conclusions using either implementation.

(b) Provide recommendations on how an aspiring baseball player should approach hitting. You may want to consider success metrics like hits where hits = 1B + 2B + 3B + HR, or weighted hits where weighted hits = 1B + 2×2B + 3×3B + 4×HR. Note these metrics are conditional on a ball being put into play in the context of this analysis.

Solution 1

```
library(VGAM)
library(nnet)
library(tidyverse)
setwd('/Users/diptarka/Documents/GitHub/stat528resources/notes/5_multinomial') ## set your own WD
bball <- read.csv("bball.csv")
bball$events <- as.factor(bball$events)
```

```
system.time(mod_vgam_small <- vglm(events ~ launch_speed + launch_angle + spray_angle +
                        I(launch_angle^2) ,
                family=multinomial, data=bball))
```

(a)

```
##    user  system elapsed
##   6.155   1.092   7.267
```

```
system.time(mod_nnet_small <- multinom(events ~ launch_speed + launch_angle + spray_angle +
               I(launch_angle^2) , trace = F,
             data=bball, maxit = 1e3))
```

```
##    user  system elapsed
##   2.196   0.013   2.214
```

```
(mod_vgam_small)
```

```
##
## Call:
## vglm(formula = events ~ launch_speed + launch_angle + spray_angle +
##     I(launch_angle^2), family = multinomial, data = bball)
##
##
## Coefficients:
##       (Intercept):1       (Intercept):2       (Intercept):3       (Intercept):4
##        -0.992044856        -8.700016742       -13.901903740       -69.069422845
##      launch_speed:1      launch_speed:2      launch_speed:3      launch_speed:4
##         0.006030511         0.066921071         0.088201226         0.416300891
##      launch_angle:1      launch_angle:2      launch_angle:3      launch_angle:4
##         0.015388970         0.134974932         0.183316005         1.847503026
##       spray_angle:1       spray_angle:2       spray_angle:3       spray_angle:4
##        -0.002331842        -0.009296012         0.017281109        -0.007929848
## I(launch_angle^2):1 I(launch_angle^2):2 I(launch_angle^2):3 I(launch_angle^2):4
##        -0.001410135        -0.003499897        -0.003941694        -0.030299742
##
## Degrees of Freedom: 200000 Total; 199980 Residual
## Residual deviance: 70874.6
## Log-likelihood: -35437.3
##
## This is a multinomial logit model with 5 levels
```

```
(mod_nnet_small)
```

```
## Call:
## multinom(formula = events ~ launch_speed + launch_angle + spray_angle +
##     I(launch_angle^2), data = bball, trace = F, maxit = 1000)
##
## Coefficients:
##     (Intercept) launch_speed launch_angle  spray_angle I(launch_angle^2)
## b2   -7.7079358  0.060890419   0.11958334 -0.006963959      -0.002089705
## b3  -12.9104603  0.082174634   0.16795879  0.019612921      -0.002532400
## b4  -68.0764825  0.410266946   1.83207561 -0.005598212      -0.028888942
## out   0.9920324 -0.006030361  -0.01538899  0.002331716       0.001410122
##
## Residual Deviance: 70874.6
## AIC: 70914.6
```

```
system.time(mod_vgam <- vglm(events ~ launch_speed + launch_angle + spray_angle +
                        I(launch_angle^2) + I(spray_angle^2) + I(spray_angle^3) + I(spray_angle^4) +
                        I(spray_angle^5) + I(spray_angle^6) + I(spray_angle*launch_angle) +
                          I(spray_angle*launch_speed) + I(launch_angle*launch_speed),
                 family=multinomial, data=bball))
```

```
##    user  system elapsed
##  16.684   2.311  19.080
```

```r
system.time(mod_nnet <- multinom(events ~ launch_speed + launch_angle + spray_angle +
               I(launch_angle^2) +
               I(spray_angle^2) + I(spray_angle^3) + I(spray_angle^4) +
               I(spray_angle^5) + I(spray_angle^6) +
               I(spray_angle*launch_angle) + I(spray_angle*launch_speed) +
               I(launch_angle*launch_speed),
             data=bball, maxit = 1e3, trace = F))
```

```
##    user  system elapsed
##  51.578   0.133  52.079
```

```r
(mod_vgam)
```

```
##
## Call:
## vglm(formula = events ~ launch_speed + launch_angle + spray_angle +
##     I(launch_angle^2) + I(spray_angle^2) + I(spray_angle^3) +
##     I(spray_angle^4) + I(spray_angle^5) + I(spray_angle^6) +
##     I(spray_angle * launch_angle) + I(spray_angle * launch_speed) +
##     I(launch_angle * launch_speed), family = multinomial, data = bball)
##
##
## Coefficients:
##                    (Intercept):1                    (Intercept):2
##                    -1.020522e+00                    -8.758315e+00
##                    (Intercept):3                    (Intercept):4
##                    -9.471725e+00                    -8.200502e+01
##                   launch_speed:1                   launch_speed:2
##                     7.481784e-03                     5.941133e-02
##                   launch_speed:3                   launch_speed:4
##                     4.154721e-02                     4.736920e-01
##                   launch_angle:1                   launch_angle:2
##                     4.413206e-02                    -3.720448e-02
##                   launch_angle:3                   launch_angle:4
##                    -1.407160e-01                     2.101850e+00
##                    spray_angle:1                    spray_angle:2
##                    -1.719423e-02                    -1.982876e-02
##                    spray_angle:3                    spray_angle:4
##                     2.235206e-02                     1.609870e-02
##              I(launch_angle^2):1              I(launch_angle^2):2
##                    -1.432220e-03                    -4.205534e-03
##              I(launch_angle^2):3              I(launch_angle^2):4
##                    -4.513380e-03                    -3.648606e-02
##               I(spray_angle^2):1               I(spray_angle^2):2
##                    -1.614774e-04                    -2.375574e-03
##               I(spray_angle^2):3               I(spray_angle^2):4
##                    -3.395291e-03                     2.587399e-03
##               I(spray_angle^3):1               I(spray_angle^3):2
##                    -5.569724e-06                    -4.859610e-07
##               I(spray_angle^3):3               I(spray_angle^3):4
##                     1.242708e-06                     5.779472e-06
##               I(spray_angle^4):1               I(spray_angle^4):2
```

```
##                   1.736069e-08                   2.809633e-06
##              I(spray_angle^4):3             I(spray_angle^4):4
##                   3.142074e-06                  -1.554782e-07
##              I(spray_angle^5):1             I(spray_angle^5):2
##                   5.978194e-10                  -1.496561e-09
##              I(spray_angle^5):3             I(spray_angle^5):4
##                  -3.256414e-10                  -1.948870e-09
##              I(spray_angle^6):1             I(spray_angle^6):2
##                   2.375031e-12                  -5.001047e-10
##              I(spray_angle^6):3             I(spray_angle^6):4
##                  -5.582904e-10                  -5.527105e-11
##  I(spray_angle * launch_angle):1  I(spray_angle * launch_angle):2
##                   2.590185e-04                   3.545018e-04
##  I(spray_angle * launch_angle):3  I(spray_angle * launch_angle):4
##                  -1.616642e-04                  -1.785922e-04
##  I(spray_angle * launch_speed):1  I(spray_angle * launch_speed):2
##                   2.204672e-04                   1.171823e-04
##  I(spray_angle * launch_speed):3  I(spray_angle * launch_speed):4
##                  -6.552925e-05                  -2.097509e-04
## I(launch_angle * launch_speed):1 I(launch_angle * launch_speed):2
##                  -3.362434e-04                   2.225316e-03
## I(launch_angle * launch_speed):3 I(launch_angle * launch_speed):4
##                   3.708676e-03                   1.309787e-03
##
## Degrees of Freedom: 200000 Total; 199948 Residual
## Residual deviance: 66221.19
## Log-likelihood: -33110.59
##
## This is a multinomial logit model with 5 levels
```

```
(mod_nnet)
```

```
## Call:
## multinom(formula = events ~ launch_speed + launch_angle + spray_angle +
##     I(launch_angle^2) + I(spray_angle^2) + I(spray_angle^3) +
##     I(spray_angle^4) + I(spray_angle^5) + I(spray_angle^6) +
##     I(spray_angle * launch_angle) + I(spray_angle * launch_speed) +
##     I(launch_angle * launch_speed), data = bball, maxit = 1000,
##     trace = F)
##
## Coefficients:
##     (Intercept) launch_speed launch_angle spray_angle I(launch_angle^2)
## b2  -0.02791403 -0.027361743  -0.29091998  0.03228890      -0.003495831
## b3  -0.00433721 -0.043320925  -0.06957590 -0.01342535      -0.003239594
## b4  -0.00927899 -0.166379000  -0.19583620  0.04830452      -0.018870088
## out  0.03767415  0.002900458  -0.01740848  0.01933244       0.001518799
##     I(spray_angle^2) I(spray_angle^3) I(spray_angle^4) I(spray_angle^5)
## b2     -0.0028247538     3.063766e-06     3.216421e-06    -2.080739e-09
## b3     -0.0054289046    -1.169831e-05     4.848559e-06     6.946726e-09
## b4      0.0002672423     1.948235e-05     9.043068e-07    -7.492347e-09
## out     0.0002288462     5.557738e-06    -4.262879e-08    -6.444846e-10
##     I(spray_angle^6) I(spray_angle * launch_angle)
## b2     -6.154897e-10                  0.0001043338
## b3     -1.024011e-09                 -0.0003091751
```

```
## b4     -2.340498e-10                 -0.0002591580
## out     2.578481e-13                 -0.0002477311
##     I(spray_angle * launch_speed) I(launch_angle * launch_speed)
## b2                  -0.0004437096                   4.995877e-03
## b3                   0.0003693118                   2.635054e-03
## b4                  -0.0006346740                   1.356923e-02
## out                 -0.0002428783                   6.756059e-06
##
## Residual Deviance: 69180.61
## AIC: 69284.61
```

Similarity:

- `vgam` and `nnet` are just different implementations of the the same underlying theoretical model. So they should give the same fitted coefficients and deviance and everything, which is the case for the smaller models fitted above, `mod_vgam_small` and `mod_nnet_small`.

Differences:

- Two implementations of models pick different baseline category. For model fitted by `vgam`, the chosen baseline category is `out`, while in model fitted by `nnet`, the chosen baseline category is `b1`. Considering this difference, we can notice that two smaller models actually give identical coefficients.
- However, the fitting results of the larger model by two implementations completely disagree. Also, the estimated standard errors of coefficients of smaller models are slightly different. This is probably caused by different optimization algorithms. `vgam` uses IRLS, while `nnet` uses BFGS. We can see this from the number of iterations of two models.
- The fitting time of `nnet` is much longer than `vgam` for large model, while for smaller model `nnet` is faster. This on the other hand may be caused by different underlying structure of two packages. `nnet` uses neural network to represent the model, of which the number of learnable parameters will grow at a faster speed as the model gets larger.

Interesting findings(according to summary table of `mod_vgam`):

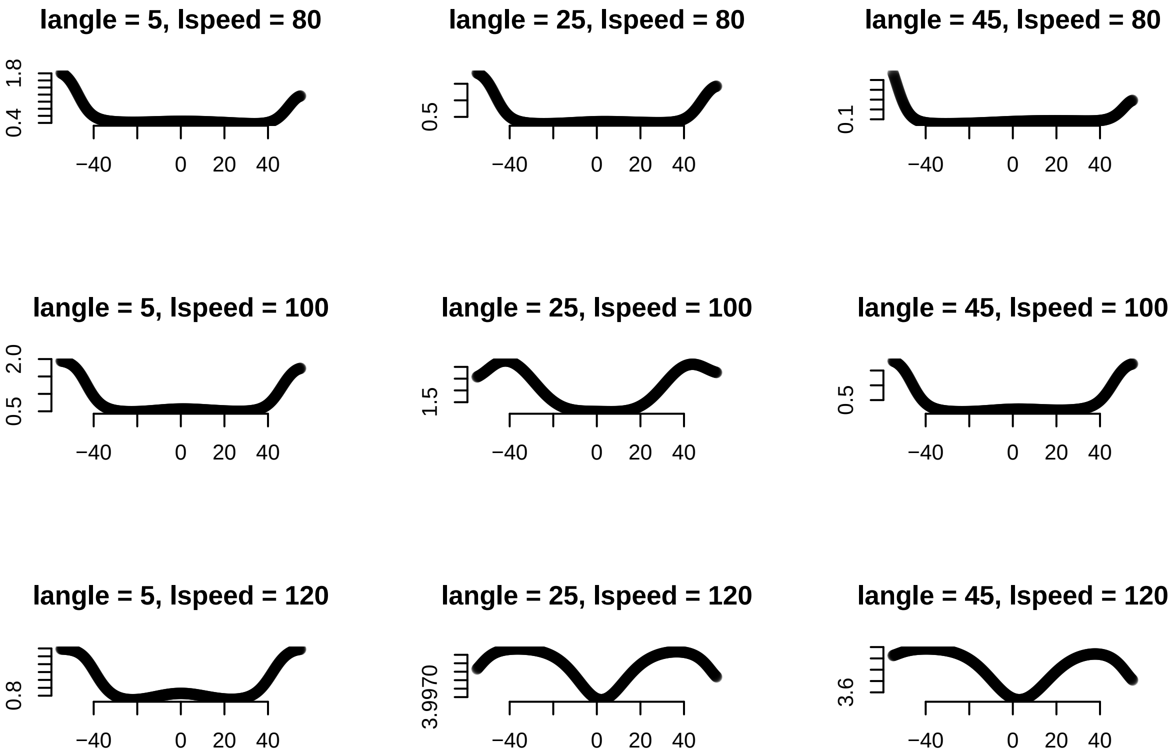
- Note that the estimated coefficients of `launch_speed` for HR has considerably greater magnitude than that of 1B, 2B, 3B. This probably means that launch speed is more crucial for a player to hit a HR. That is to say, good spray angle and launch angle may be sufficient to hit 1B, 2B and 3B, but for a HR, great launch speed is a must.
- For higher order(greater than 2) polynomials of `spray_angle`, only few of them are significant. In particular, for HR, none of the higher order polynomials of `spary_angle` is significant for HR. On the other hand, all coefficients of even order polynomials of `spray_angle` are significant for 2B and 3B. Our conclusion from last bullet point is supported by these findings: good angles are important for 2B and 3B, but for HR, launch speed is the one that makes difference.
- More interestingly, as mentioned in the last point, basically only even order polynomials of `spray_angle` display significance. Does this mean that actually only the magnitude is needed for `spray angle`? The reason we include `spray_angle` up to order of 6 is that we want to account for the positions of opponent catchers. But this model is probably saying that this is not necessary.

(b) Calculate the weighted hits as success metric.

```r
weighted_hits <- function(lspeed, langle) {
  ## obtain predictions
  new_data = data.frame(spray_angle = seq(-55,55, by = 0.1),
              launch_speed = lspeed,
              launch_angle = langle)
  pred <- predict(mod_vgam, newdata = new_data, type = 'response')
  weighted_hits <- pred[,1] + 2*pred[,2] + 3*pred[,3] + 4*pred[,4]

  ## Make plot
  plot.new()
  title(paste0( "langle = ",langle, ", lspeed = ", lspeed))
  plot.window(xlim = c(-55,55), ylim = c(min(weighted_hits), max(weighted_hits)))
  points(new_data$spray_angle, weighted_hits, pch = 19, col = rgb(0,0,0,alpha=0.2))
  axis(1)
  axis(2)
}
```

Plot weighted hits against spray angle, under different combinations of launch speed and launch angle.

```r
par(mfrow = c(3,3))
weighted_hits(80, 5)
weighted_hits(80, 25)
weighted_hits(80, 45)
weighted_hits(100, 5)
weighted_hits(100, 25)
weighted_hits(100, 45)
weighted_hits(120, 5)
weighted_hits(120, 25)
weighted_hits(120, 45)
```



According to plots above, we recommend a player to - hit the ball as hard as possible(so launch speed will be fast), - keep the launch angle in the positive middle range, i.e. 20-30 degrees - and try to hit the ball around side line.

Problem 2: A study of factors affecting alcohol consumption measures the response variable with the scale (abstinence, a drink a day or less, more than one drink a day). For a comparison of two groups while adjusting for relevant covariates, the researchers hypothesize that the two groups will have about the same prevalence of abstinence, but that one group will have a considerably higher proportion who have more than one drink a day. Even though the response variable is ordinal, explain why a cumulative logit model with proportional odds structure may be inappropriate for this study.

Solution 2:

The study has two groups let us assume they are G_1 and G_2.

Now the researchers hypothesize that the two groups have about the same prevalance of abstinence which means that

$$\pi_1(G_1) \approx \pi_1(G_2)$$

Also they hypothesize that one group will have a considerably higher proportion who have more than one drink a day i.e.

$$\pi_3(G_1) > \pi_3(G_2)$$

Now in the cumulative logit model with proportional odds structure we have the property that

$$logit(P(Y \leq j|G_1)) - logit(P(Y \leq j|G_2)) \text{ does not depend on j}$$

For j = 1 we have

$$logit(P(Y \leq 1|G_1)) - logit(P(Y \leq 1|G_2)) = log\left(\frac{\pi_1(G_1)}{1-\pi_1(G_1)}\right) - log\left(\frac{\pi_1(G_2)}{1-\pi_1(G_2)}\right) \approx 0$$

But for j = 3

$$logit(P(Y \leq 3|G_1)) - logit(P(Y \leq 3|G_2)) = log\left(\frac{\pi_3(G_1)}{1-\pi_3(G_1)}\right) - log\left(\frac{\pi_3(G_2)}{1-\pi_3(G_2)}\right) \neq 0$$

Thus the cumulative logit model with proportional odds structure is not appropriate here. ###

Problem 3: Refer to the table below:

		Belief in Heaven		
Race	Gender	Yes	Unsure	No
Black	Female	88	16	2
	Male	54	17	5
White	Female	397	141	24
	Male	235	189	39

(a) Fit the model

$$\log(\pi_j/\pi_3) = \alpha_j + \beta_j^G x_1 + \beta_j^R x_2, \qquad j = 1, 2.$$

(b) Find the prediction equation for $\log(\pi_1/\pi_2)$.

(c) Treating belief in heaven as ordinal fit and interpret a cumulative logit model and a cumulative probit model. Compare results and state interpretations in each case.

Solution 3

(a) Want to fit the model

$$\log(\pi_j/\pi_3) = \alpha_j + \beta_j^G x_1 + \beta_j^R x_2$$

We consider belief in heaven to be a nominal variable and create a dataset which reflects the table

```
nominal_heaven = data.frame("Race" = c("Black","Black","White","White"),"Gender" =c("Female","Male","Fe
nominal_heaven
```

```
##    Race Gender Belief_Yes Belief_Unsure Belief_No
## 1 Black Female         88            16         2
## 2 Black   Male         54            17         5
## 3 White Female        397           141        24
## 4 White   Male        235           189        39
```

Now that we have the data we fit the baseline-category logistic model for the multinomial response data

```
library(VGAM)
mod_nom <- vglm(cbind(Belief_Yes, Belief_Unsure, Belief_No) ~ Race + Gender,
family=multinomial, data=nominal_heaven)
summary(mod_nom)
```

```
##
## Call:
## vglm(formula = cbind(Belief_Yes, Belief_Unsure, Belief_No) ~
##     Race + Gender, family = multinomial, data = nominal_heaven)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1   3.5318     0.4203   8.403  < 2e-16 ***
## (Intercept):2   1.7026     0.4483   3.798 0.000146 ***
## RaceWhite:1    -0.7031     0.4113  -1.710 0.087349 .
## RaceWhite:2     0.1056     0.4384   0.241 0.809613
## GenderMale:1   -1.0435     0.2587  -4.034 5.48e-05 ***
## GenderMale:2   -0.2545     0.2691  -0.946 0.344277
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,1]/mu[,3]), log(mu[,2]/mu[,3])
##
## Residual deviance: 0.6903 on 2 degrees of freedom
##
## Log-likelihood: -19.4657 on 2 degrees of freedom
##
## Number of Fisher scoring iterations: 3
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):1'
##
##
## Reference group is level  3  of the response
```

Thus $\alpha_1 = 3.5318, \alpha_2 = 1.7026, \beta_1^G = -1.0435, \beta_2^G = -0.2545, \beta_1^R = -0.7031, \beta_2^R = 0.1056$

(b) Find the prediction equation for $\log(\pi_1/\pi_2)$

We use the fact that

$$\log(\pi_1/\pi_2) = \log(\pi_1/\pi_3) - \log(\pi_1/\pi_3)$$

Thus the prediction equation for $\log(\pi_1/\pi_2)$ is

$$
\begin{aligned}
log(\pi_1/\pi_2) &= log(\pi_1/\pi_3) - log(\pi_1/\pi_3) \\
&= \alpha_1 - \alpha_2 + (\beta_1^G - \beta_2^G)x_1 + (\beta_1^R - \beta_2^R)x_2 \\
&= (3.5318 - 1.7026) + (-1.0435 + 0.2545)x_1 + (-0.7031 - 0.1056)x_2 \\
&= 1.8292 - 0.789x_1 - 0.8087x_2
\end{aligned}
$$

(c) Treating belief in heaven as ordinal fit and interpret a cumulative logit model and a cumulative probit model. Compare results and state interpretations in each case.

We first create the data treating "Belief in heaven" as an ordinal variable.

```
rep.row<-function(x,n){
   matrix(rep(x,each=n),nrow=n)
}

data_heaven = rbind(rep.row(c("B","F",1),88),rep.row(c("B","F",2),16),rep.row(c("B","F",3),2),rep.row(c


colnames(data_heaven) = c("Race","Gender","Belief")

data_heaven = as.data.frame(data_heaven)
head(data_heaven)
```

```
##   Race Gender Belief
## 1    B      F      1
## 2    B      F      1
## 3    B      F      1
## 4    B      F      1
## 5    B      F      1
## 6    B      F      1
```

```
data_heaven$Race = factor(data_heaven$Race)
data_heaven$Gender = factor(data_heaven$Gender)
data_heaven$Belief = factor(data_heaven$Belief,ordered = T )
```

We fit the cumulative logit model i.e.

$G^{-1}(P(Y \leq j|x)) = \alpha_j - x^T\beta$ {where G^{-1} is the logit function}

```
mod <- vglm(Belief ~ Race+Gender, family=propodds(reverse=FALSE),
data=data_heaven)
summary(mod)
```

```
##
## Call:
## vglm(formula = Belief ~ Race + Gender, family = propodds(reverse = FALSE),
##     data = data_heaven)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept):1   1.6379     0.1910   8.576  < 2e-16 ***
## (Intercept):2   3.9138     0.2259  17.329  < 2e-16 ***
## RaceW          -0.7685     0.1911  -4.021 5.80e-05 ***
## GenderM        -0.8217     0.1215  -6.765 1.33e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]), logitlink(P[Y<=2])
##
## Residual deviance: 1893.45 on 2410 degrees of freedom
##
## Log-likelihood: -946.7251 on 2410 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
##     RaceW   GenderM
## 0.4637078 0.4396769
```

We fit the cumulative probit model i.e.

$G^{-1}(P(Y \leq j|x)) = \alpha_j - x^T\beta$ {where G^{-1} is the probit function}

```
mod_prob <- vglm(Belief ~ Race+Gender, family=cumulative(link="probitlink",parallel=TRUE),
data=data_heaven)
summary(mod_prob)
```

```
##
## Call:
## vglm(formula = Belief ~ Race + Gender, family = cumulative(link = "probitlink",
##     parallel = TRUE), data = data_heaven)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept):1  0.95883    0.10703   8.959  < 2e-16 ***
## (Intercept):2  2.21185    0.12050  18.356  < 2e-16 ***
## RaceW         -0.43055    0.10809  -3.983 6.79e-05 ***
## GenderM       -0.47935    0.07147  -6.707 1.99e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: probitlink(P[Y<=1]), probitlink(P[Y<=2])
##
## Residual deviance: 1896.014 on 2410 degrees of freedom
##
## Log-likelihood: -948.0071 on 2410 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## No Hauck-Donner effect found in any of the estimates
##
```

```
##
## Exponentiated coefficients:
##     RaceW   GenderM
## 0.6501506 0.6191833
```

We can see that the log-likelihood value (and thus the AIC) and the residual deviance for both the models is approximately the same and thus both the models have a similar performance despite the underlying models being different. Also note that the signs of the estimates of the coefficients are also the same in both models which means that the variables have the same relationship with the response in both models.

Problem 4: Suppose that you have a coin that when flipped has a probability $0 < p < 1$ of landing heads, and that we know nothing about p. Suppose that you flip the coin four times and all four flips resulted in heads. Derive the MLE of p and the MLE of $\mathrm{Var}(Y_i)$ under the standard Bernoulli model. Now, for some error tolerance $0 < \alpha < 1$, derive a valid one-sided confidence interval for p making use of the statement $\mathbb{P}\left(\sum_{i=1}^{4} y_i = 4\right)$.

Solution 4:

The log-likelihood of the Bernoulli distribution is

$$l(p|y_i) = \log p \sum y_i + (n - \sum y_i)\log(1-p)$$

To find MLE,

$$\begin{aligned}
\log \hat{p} \sum y_i + (n - \sum y_i)\log(1-\hat{p}) &= 0 \\
\implies \frac{\sum y_i}{\hat{p}} - \frac{(n - \sum y_i)}{1-\hat{p}} &= 0 \\
\implies \hat{p} = \frac{1}{n}\sum y_i
\end{aligned}$$

Since we get four heads in four tosses $\sum y_i = 4$ and n = 4.

$$\implies \hat{p} = 1$$

Now $Var(Y_i) = p(1-p) \implies Va\hat{r}(Y_i) = \hat{p}(1-\hat{p})$

$$\implies \hat{V}ar(Y_i) = 0$$

This means that the space of γ such that γ belongs to the null space of $Var(Y_i)$ is $\mathbb{R}$

Thus the lower boundary of the Confidence interval for p is

$$\min_{P_p(\sum y_i = 4) \geq \alpha} p$$

Now

$$\begin{aligned}
P_p(\sum y_i = 4) \geq \alpha &\implies p^{\sum y_i}(1-p)^{n - \sum y_i} \geq \alpha \\
&\implies p^4(1-p)^{4-4} \geq \alpha
\end{aligned}$$

Now since the log likelihood is a concave function the min p which satisfies the constraint will satisfy

$$p^4 = \alpha$$

$$\implies p = \alpha^{1/4}$$

Thus the $100(1-\alpha)\%$ one sided confidence interval is

$$CI = (\alpha^{1/4}, 1)$$

Problem 5: Complete the following with respect to the `endometrial` example:

(a) Write your own Fisher scoring algorithm for this example. Argue that $\hat{\beta}$ diverges in some sense as the iterations of your algorithm increase.

(b) Show that the log likelihood has an asymptote in $\|\beta\|$.

(c) Code the likelihood function for this dataset, pick a value of $\tilde{\beta}$ that is in the LCM, find an eigenvector of estimated Fisher information η such that the likelihood asymptotes, and then show that the likelihood asymptotes in $\tilde{\beta} + s\eta$ as $s \to \infty$.

(d) Explain why the likelihood asymptotes in $\tilde{\beta} + s\eta$ as $s \to \infty$.

Solution 5:

(a) Write your own Fisher scoring algorithm for this example. Argue that $\hat{\beta}$ diverges in some sense as the iterations of your algorithm increase.

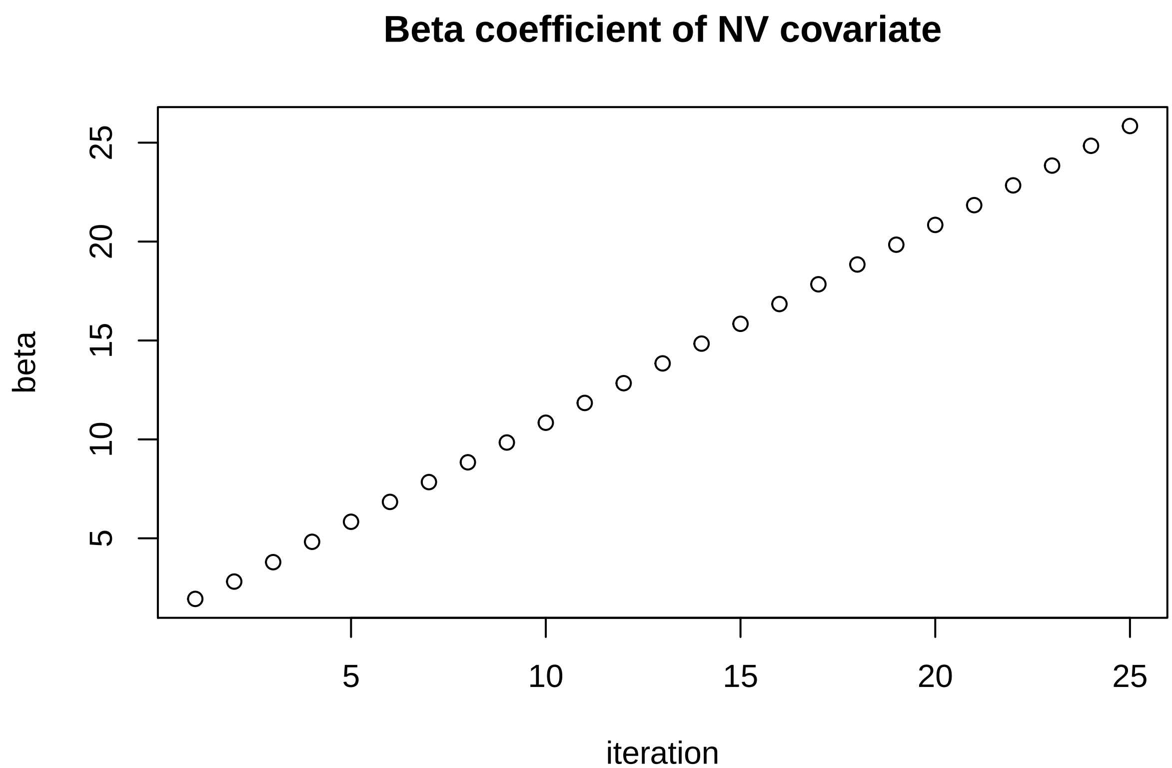
```
#Creating a function to compute the log-likelihood
log_lik = function(X,Y,beta)
{
  t(Y)%*%X%*%beta - sum(log(1 + exp(X%*%beta)))
}
```

```
library(enrichwith)
data(endometrial)

#Creating the model matrix
X = model.matrix(HG ~ .,data = endometrial)
n = nrow(X)
p = ncol(X)
Y = endometrial$HG

# Initializing the beta
beta = matrix(rep(0,p))
beta_list = NULL
#Running the Fisher scoring iterations
for(t in 1:25)
{
```

```
  pi =  exp(X%*%beta)/(1+exp(X%*%beta))
  W = diag(c(pi*(1-pi)))
  beta = beta + solve(t(X) %*% W %*% X)%*%t(X)%*%(Y - pi)
  beta_list = cbind(beta_list,beta)
}
```

To show that the $\hat{\beta}$ diverges in some sense we plot the $\hat{\beta}$ corresponding to the NV variable over the iterations

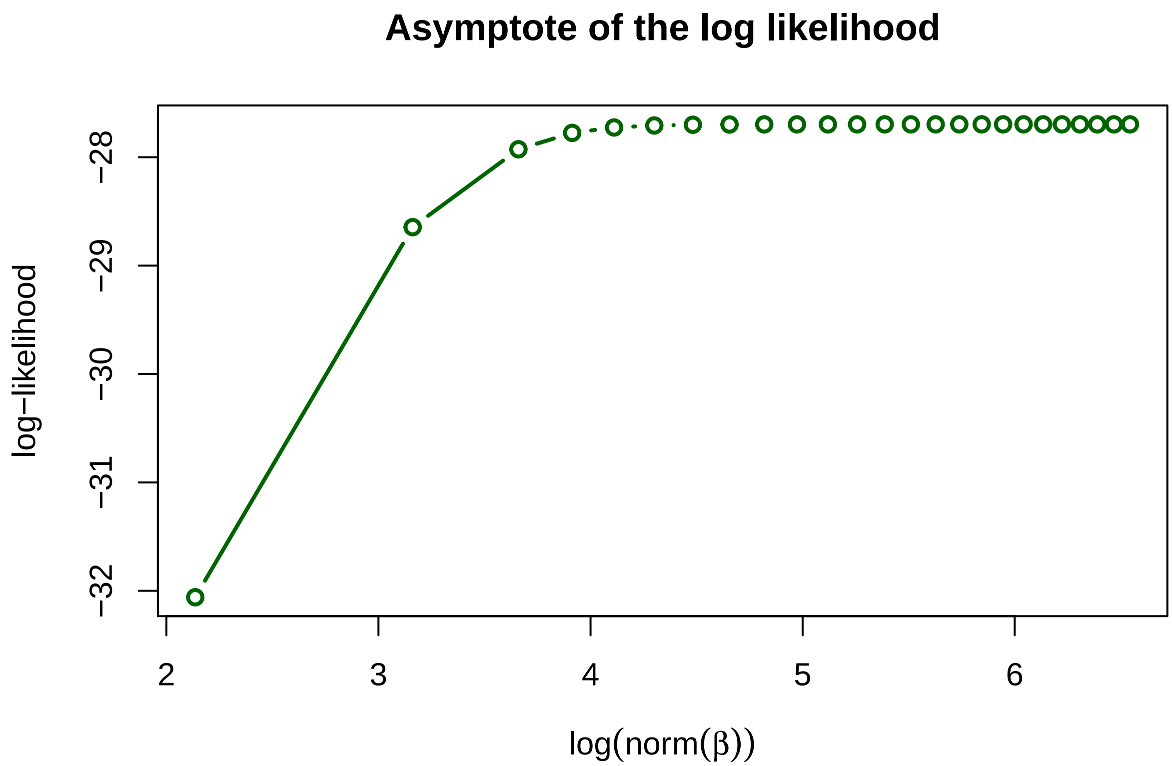
```
plot(1:25,beta_list[2,],main = "Beta coefficient of NV covariate",xlab = "iteration",ylab = "beta")
```



Clearly it diverges.

(b) Show that the log likelihood has an asymptote in $\|\beta\|$

```
#Computing the log likelihood for each of the beta values that we get
yvalues = apply(beta_list,2,function(t) log_lik(X,Y,t))

#Computing the norm of the beta values
xvalues = log(apply(beta_list,2,function(t) sum(t^2)))

#Plotting
plot(xvalues,yvalues,ty = "b",lwd = 2,col = "darkgreen", main = "Asymptote of the log likelihood",xlab =
```



Clearly it aysmptotes in $\|\beta\|$

(c) Code the likelihood function for this dataset, pick a value of $\hat{\beta}$ that is in the LCM, extract the null eigen vector of estimated Fisher information η, and then show that the likelihood asymptotes in $\beta + s\eta$ as $s \to \infty$.

```
#Creating a logistic model model for the endometrial data
mod <- glm(HG ~ ., family = "binomial",
control = list(maxit = 25, epsilon = 1e-100),data = endometrial)
summary(mod)
```

```
##
## Call:
## glm(formula = HG ~ ., family = "binomial", data = endometrial,
##     control = list(maxit = 25, epsilon = 1e-100))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5014  -0.6411  -0.2943   0.0000   2.7278
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.305e+00  1.637e+00   2.629 0.008563 **
## NV           2.619e+01  9.368e+04   0.000 0.999777
## PI          -4.218e-02  4.433e-02  -0.952 0.341333
```

```
## EH          -2.903e+00  8.456e-01  -3.433 0.000597 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 104.903  on 78  degrees of freedom
## Residual deviance:  55.393  on 75  degrees of freedom
## AIC: 63.393
##
## Number of Fisher Scoring iterations: 25
```

We can see that the Fisher Scoring algorithm goes through all the 25 iterations (which was specified as the max number of iterations). We have seen from part (b) that after 25 fisher scoring iteration the likelihood has already converged and thus the null space of the fisher scoring matrix obtained using the parameters of the OM at this stage estimate the null space of the fisher scoring matrix of the LCM well.

```r
#Computing the beta belonging to the LCM
beta_tilde = mod$coefficients
beta_tilde
```

```
## (Intercept)          NV          PI          EH
##   4.3045178  26.1855559  -0.0421834  -2.9026056
```

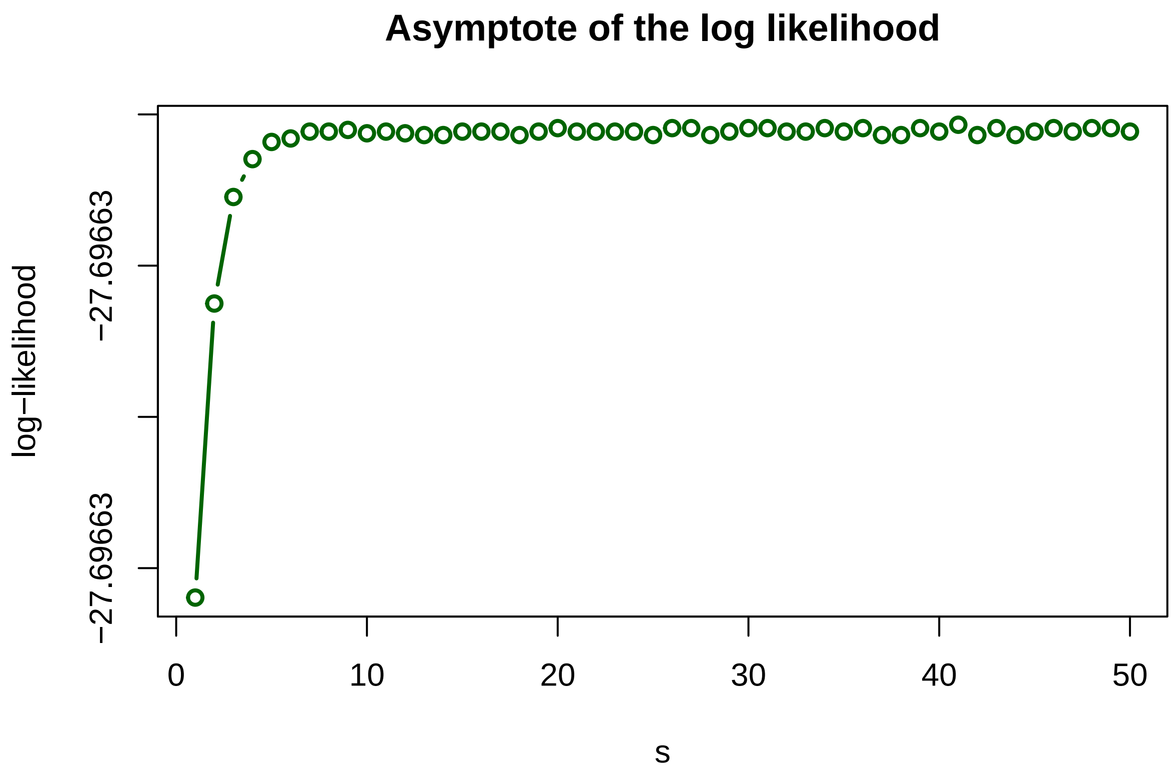
```r
#Finding the null eigenvector of the Fisher scoring matrix
invFI <- vcov(mod)
FI <- solve(invFI)
eig = eigen(FI)
eig
```

```
## eigen() decomposition
## $values
## [1] 3.068079e+03 7.175314e+00 3.069389e-01 1.138352e-10
##
## $vectors
##               [,1]          [,2]          [,3]          [,4]
## [1,] -4.914836e-02  4.272808e-01  9.027821e-01  1.284014e-11
## [2,] -9.759971e-13 -1.266325e-11  2.016320e-11 -1.000000e+00
## [3,] -9.962646e-01 -8.522654e-02 -1.390049e-02  1.771472e-12
## [4,] -7.100158e-02  9.000931e-01 -4.298734e-01 -1.999677e-11
```

Clearly the last eigen value is 0 thus the corresponding eigen vector belongs to the null space of the Fisher information matrix. We can see that the null vector is $-e_2$ i.e $(0,-1,0,0)$. Thus we can consider $\eta = (0,1,0,0)$

```r
eta = -eig$vectors[,4]

#Considering s from 1 to 50
s = c(1:50)

#Creating a list of beta of the form beta_tilde + s*eta
vals = matrix(unlist(lapply(s,function(t) beta_tilde + t*eta)),nrow = 4)
```

```r
#Computing the loglik values for these beta
yvalues = apply(vals,2,function(t) log_lik(X,Y,t))

#Plotting the loglik values against s
plot(s,yvalues,ty = "b",lwd = 2,col = "darkgreen", main = "Asymptote of the log likelihood",xlab = "s",
```



Again clearly it asymptotes as $s \to \infty$

(d) LCM is the OM with lost dimensions. In other words, the sub-model canonical statistics of LCM is restricted to a hyper-plane in the support set. Since sub-model canonical statistics of LCM is constrained to the hyper-plane, it can not vary along the direction that is orthogonal to the hyper-plane, hence vectors η that are orthogonal to the hyper-plane span the null space of the Fisher information matrix of LCM. Recall that null space of Fisher information matrix can be approximated by Fisher information matrix of OM, so eigen vectors of OM are approximately orthogonal to the hyper-plane. Therefore, $\tilde{\beta} + s\eta$ is gradually moving $\tilde{\beta}$ towards to the orthogonal direction of hyper-plane. But sub-model canonical statistics can not vary along that direction. So, as $s \to \infty$, sub-model canonical statistics will gradually stop moving, leading to asymptote of likelihood.

Problem 6: Summarise the Firth approach mentioned in Section 7.4.7 and 7.4.8 of Agresti. Compare and contrast the Firth approach with the direct MLE approach outlined in the complete separation notes. What are the strengths and weaknesses of each approach?

Solution 6:

In the Firth approach we penalise the log-likelihood function to ensure that the MLE always exists. The penalized log-likelihood function utilizes the determinant of the information matrix $\mathcal{J}$,

$$L^*(\beta) = L(\beta) + \frac{1}{2}\log|\mathcal{J}|$$

It turns out that this approach coincides with the Bayesian approach with Jeffrey's prior.

Comparison:

- Even under the case of complete and quasi-separation, Firth's approach can still give finite and unique estimates of coefficients with decent certainty. On the other hand, the direct MLE approach(OM and LCM) can only give unbouded estimate intervals for separable variables.
- However, the introduction of penalization makes Firth's approach tend to give larger estimated coefficients than MLE, leading to inaccurate estimation under certain cases. As for the direct MLE approach, estimated coefficients of non-separable covariates are generally reliable.
- Firth's approach uses second order approximation, hence can reduce bias to order $1/n^2$, while direct MLE only uses first order approximation, which has order $1/n$ bias.
- Note that Firth's approach actually falls into the category of Bayesian approaches, which come with the problem of choosing priors. It is shown that different priors have different merits, and can all give reasonable results. These facts make such approaches less objective. However, the frequentist approach, direct MLE, is always objective.

Problem 7: Use `glmdr` software to analyze the `catrec.txt` data using Poisson regression. Specifically, fit a third order model and provide confidence intervals for all mean-value parameter estimates, both one-sided intervals for responses that are constrained on the boundary and two-sided intervals for responses that are unconstrained. Also verify that the third order model is appropriate.

Solution 7:

part a Fisher scoring algorithm:

```
library(enrichwith)
data(endometrial)

 m = glm(HG ~ ., data = endometrial, family = "binomial",
 x = TRUE, y = TRUE)
 summary(m)
```

```
Call:
glm(formula = HG ~ ., family = "binomial", data = endometrial,
    x = TRUE, y = TRUE)
```

```
Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.50137  -0.64108  -0.29432   0.00016   2.72777

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    4.30452    1.63730   2.629 0.008563 **
NV            18.18556 1715.75089   0.011 0.991543
PI            -0.04218    0.04433  -0.952 0.341333
EH            -2.90261    0.84555  -3.433 0.000597 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 104.903  on 78  degrees of freedom
Residual deviance:  55.393  on 75  degrees of freedom
AIC: 63.393

Number of Fisher Scoring iterations: 17
```

```r
x=endometrial[,-4]
x$intercept=rep(1,nrow(x))
x=as.matrix(x)
y=endometrial$HG

d=dim(x)
p=d[2]
n=d[1]
mustart = y + 0.1
thetastart = log(mustart)
b.init = rep(0,p)
eta <- x %*% b.init #predicted logit
mu <- plogis(eta)
vr <- (mu * (1 - mu))
wts <- diag(as.vector(vr)) #create weight matrix
d2 <- t(x) %*% wts %*% x #the Fisher information matrix
u1 <- t(x) %*% (y - mu) #the score function
iter <- 25 #max number of iterations
tol <- 1e-14 #tolerance

for (i in 1:iter){
  beta_1 <- b.init + solve(d2) %*% u1 #update beta
  #beta_1 <- b.init + chol2inv(chol(d2)) %*% u1 #update beta avoiding inversion

  #if(any(abs((beta_1 - b.init) / b.init) < tol)) break #stop loop if no change

  eta <- x %*% beta_1 #predicted logit
  mu <- plogis(eta) #this is just pp(eta)
  vr <- mu * (1 - mu) #variance

  wts <- diag(as.vector(vr)) #putting in a weight matrix
  d2 <- t(x) %*% wts %*% x #information matrix
  u1 <- t(x) %*% (y - mu) #score function
```

```r
  b.init <- beta_1
  print(beta_1)
  cat(i, "::") #show iteration number
}
```

```
                  [,1]
NV          1.93677380
PI         -0.01819958
EH         -1.32579121
intercept  1.71957691
1 ::                  [,1]
NV          2.81200210
PI         -0.02992848
EH         -2.27856889
intercept  3.24074395
2 ::                  [,1]
NV          3.79409231
PI         -0.03863693
EH         -2.78647063
intercept  4.08676510
3 ::                 [,1]
NV          4.8238176
PI         -0.0418331
EH         -2.8966021
intercept  4.2903146
4 ::                  [,1]
NV          5.83635228
PI         -0.04216911
EH         -2.90246038
intercept  4.30406445
5 ::                  [,1]
NV          6.84044302
PI         -0.04218242
EH         -2.90259300
intercept  4.30448256
6 ::                  [,1]
NV          7.84191969
PI         -0.04218328
EH         -2.90260397
intercept  4.30451331
7 ::                  [,1]
NV          8.84246141
PI         -0.04218339
EH         -2.90260539
intercept  4.30451718
8 ::                 [,1]
NV          9.8426605
PI         -0.0421834
EH         -2.9026056
intercept  4.3045177
9 ::                 [,1]
NV         10.8427338
PI         -0.0421834
```

```
EH        -2.9026056
intercept  4.3045178
10 ::               [,1]
NV        11.8427607
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
11 ::               [,1]
NV        12.8427706
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
12 ::               [,1]
NV        13.8427742
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
13 ::               [,1]
NV        14.8427756
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
14 ::               [,1]
NV        15.8427761
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
15 ::               [,1]
NV        16.8427763
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
16 ::               [,1]
NV        17.8427763
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
17 ::               [,1]
NV        18.8427764
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
18 ::               [,1]
NV        19.8427764
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
19 ::               [,1]
NV        20.8427764
PI        -0.0421834
EH        -2.9026056
intercept  4.3045178
20 ::               [,1]
NV        21.8427764
```

```
PI         -0.0421834
EH         -2.9026056
intercept  4.3045178
21 ::                 [,1]
NV         22.8427764
PI         -0.0421834
EH         -2.9026056
intercept  4.3045178
22 ::                 [,1]
NV         23.8427764
PI         -0.0421834
EH         -2.9026056
intercept  4.3045178
23 ::                 [,1]
NV         24.8427764
PI         -0.0421834
EH         -2.9026056
intercept  4.3045178
24 ::                 [,1]
NV         25.8427764
PI         -0.0421834
EH         -2.9026056
intercept  4.3045178
25 ::
```

```r
se <- sqrt(diag(solve(d2))) #get standard error, inverse of information matrix
z <- beta_1 / se
p.val <- 2 * pt(-abs(z), df = Inf)
df <- data.frame(beta = beta_1, se = se, z = z, p = format(p.val, 3))
row.order=c("intercept","NV","PI","EH")
df[row.order,]
```

```
                beta           se             z            p
intercept  4.3045178 1.637299e+00  2.6290364342 0.0085627186
NV        25.8427764 1.301205e+05  0.0001986066 0.9998415349
PI        -0.0421834 4.433197e-02 -0.9515346986 0.3413330130
EH        -2.9026056 8.455516e-01 -3.4327955194 0.0005973925
```

We can observe that as the iterations of your algorithm increase, the coefficient of NV increases, whereas the coefficients of other variables converged to some value (same as what is given in the summary table of glm) after a few iterations. So, the $\hat{\beta}$ diverges in the direction corresponding to NV (0,1,0,0).

part b

```r
## likelihood asymptote
asymptote = t(sapply(1:30, function(iter){
  m_test = glm(HG ~ ., family = "binomial", data = endometrial,
               control = list(maxit = iter, epsilon = 1e-20))
  c(sqrt(log(crossprod(coef(m_test)))), logLik(m_test), coef(m_test))
}))
asymptote = as.data.frame(asymptote)
asymptote
```

```
          V1        V2 (Intercept)        NV          PI        EH
1  1.589704 -30.39406    2.090959  2.355064 -0.02213019 -1.612126
2  1.836053 -28.25689    3.599433  3.144596 -0.03327115 -2.502950
3  1.939476 -27.85263    4.203858  4.148140 -0.04030884 -2.852134
4  1.995684 -27.75239    4.300150  5.172614 -0.04205976 -2.900986
5  2.043730 -27.71703    4.304349  6.181031 -0.04217827 -2.902550
6  2.088998 -27.70412    4.304501  7.183901 -0.04218293 -2.902599
7  2.131374 -27.69938    4.304516  8.184948 -0.04218334 -2.902605
8  2.170821 -27.69764    4.304517  9.185332 -0.04218340 -2.902606
9  2.207465 -27.69700    4.304518 10.185474 -0.04218340 -2.902606
10 2.241507 -27.69677    4.304518 11.185526 -0.04218340 -2.902606
11 2.273173 -27.69668    4.304518 12.185545 -0.04218340 -2.902606
12 2.302690 -27.69665    4.304518 13.185552 -0.04218340 -2.902606
13 2.330270 -27.69664    4.304518 14.185554 -0.04218340 -2.902606
14 2.356107 -27.69663    4.304518 15.185555 -0.04218340 -2.902606
15 2.380373 -27.69663    4.304518 16.185556 -0.04218340 -2.902606
16 2.403223 -27.69663    4.304518 17.185556 -0.04218340 -2.902606
17 2.424791 -27.69663    4.304518 18.185556 -0.04218340 -2.902606
18 2.445197 -27.69663    4.304518 19.185556 -0.04218340 -2.902606
19 2.464548 -27.69663    4.304518 20.185556 -0.04218340 -2.902606
20 2.482934 -27.69663    4.304518 21.185556 -0.04218340 -2.902606
21 2.500440 -27.69663    4.304518 22.185556 -0.04218340 -2.902606
22 2.517138 -27.69663    4.304518 23.185556 -0.04218340 -2.902606
23 2.533094 -27.69663    4.304518 24.185556 -0.04218340 -2.902606
24 2.548364 -27.69663    4.304518 25.185556 -0.04218340 -2.902606
25 2.563001 -27.69663    4.304518 26.185556 -0.04218340 -2.902606
26 2.577052 -27.69663    4.304518 27.185553 -0.04218340 -2.902606
27 2.590557 -27.69663    4.304518 28.185557 -0.04218340 -2.902606
28 2.603556 -27.69663    4.304518 29.185576 -0.04218340 -2.902606
29 2.616082 -27.69663    4.304518 30.185606 -0.04218340 -2.902606
30 2.628163 -27.69663    4.304518 31.185428 -0.04218340 -2.902606
```

In the above table, we can see the result which is consistent with what we said in part (a). We can see that as the number of iterations increase, the coefficient of NV increases and others converge and hence $||\beta||$ increases. We can also notice that the log likelihood (2nd column in the above table) also converges to some value. Therefore, we can say that the log likelihood has an asymptote in $||\beta||$.

part c

```
x=endometrial[,-4]
x=cbind(rep(1,nrow(x)),x)
colnames(x)=c("intercept",colnames(x)[-1])
x=as.matrix(x)
y=endometrial$HG

#Likelihood function for the data
L=function(x,y,beta){
  exp(sum(y*log(plogis(x%*%beta))+(1-y)*log(1-plogis(x%*%beta))))
}
#Checking
beta=c(4.304518, 14.185554, -0.04218340, -2.902606)
L(x,y,beta)
```

```
[1] 9.364905e-13
```

```r
log(L(x,y,beta))
```

```
[1] -27.69664
```

```r
m_glmdr = glmdr(HG ~ ., data = endometrial, family = "binomial")
 summary(m_glmdr)
```

```
MLE exists in Barndorff-Nielsen completion
it is conditional on components of the response
corresponding to object$linearity == FALSE being
conditioned on their observed values



GLM summary for limiting conditional model


Call:
stats::glm(formula = HG ~ ., family = "binomial", data = endometrial,
    subset = c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10",
    "11", "12", "13", "14", "15", "16", "17", "18", "19", "20",
    "21", "27", "28", "29", "30", "31", "32", "33", "34", "35",
    "36", "37", "38", "39", "40", "41", "42", "43", "44", "45",
    "46", "47", "52", "53", "54", "55", "56", "57", "58", "59",
    "60", "61", "62", "63", "64", "65", "66", "67", "68", "69",
    "70", "72", "73", "74", "77", "79"), x = TRUE, y = TRUE)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5014  -0.6634  -0.3856   0.2126   2.7278

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.30452    1.63720   2.629 0.008559 **
NV                NA         NA      NA       NA
PI          -0.04218    0.04433  -0.952 0.341310
EH          -2.90261    0.84549  -3.433 0.000597 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 75.307  on 65  degrees of freedom
Residual deviance: 55.393  on 63  degrees of freedom
AIC: 61.393

Number of Fisher Scoring iterations: 5
```

```r
  m2 = update(m, subset = m_glmdr$linearity)
 summary(m2)
```

```
Call:
glm(formula = HG ~ ., family = "binomial", data = endometrial,
    subset = m_glmdr$linearity, x = TRUE, y = TRUE)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5014  -0.6634  -0.3856   0.2126   2.7278

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.30452    1.63720   2.629 0.008559 **
NV                NA         NA      NA       NA
PI          -0.04218    0.04433  -0.952 0.341310
EH          -2.90261    0.84549  -3.433 0.000597 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 75.307  on 65  degrees of freedom
Residual deviance: 55.393  on 63  degrees of freedom
AIC: 61.393

Number of Fisher Scoring iterations: 5
```

```
 beta=m2$coeff[-2] #choosing beta from LCM
(e=eigen(solve(vcov(m))))
```

```
eigen() decomposition
$values
[1] 3.068079e+03 7.175315e+00 3.069390e-01 3.396973e-07

$vectors
              [,1]          [,2]          [,3]          [,4]
[1,] -4.914836e-02  4.272808e-01  9.027821e-01  3.831803e-08
[2,] -2.909620e-09 -3.774294e-08  6.014946e-08 -1.000000e+00
[3,] -9.962646e-01 -8.522653e-02 -1.390049e-02  5.279344e-09
[4,] -7.100158e-02  9.000931e-01 -4.298735e-01 -5.962223e-08
```

```
eta1=e$vectors[which(abs(e$vectors[,4])<1e-06),4] #choosing eta

asymptote = t(sapply(1:30, function(s){
  L=L(x[,-2],y,beta+s*eta1)
  c(L, s)
}))
asymptote = as.data.frame(asymptote)
asymptote
```

```
             V1 V2
1  1.341817e-15  1
2  1.341818e-15  2
3  1.341819e-15  3
```

```
4  1.341819e-15  4
5  1.341820e-15  5
6  1.341821e-15  6
7  1.341821e-15  7
8  1.341822e-15  8
9  1.341822e-15  9
10 1.341823e-15 10
11 1.341824e-15 11
12 1.341824e-15 12
13 1.341825e-15 13
14 1.341826e-15 14
15 1.341826e-15 15
16 1.341827e-15 16
17 1.341828e-15 17
18 1.341828e-15 18
19 1.341829e-15 19
20 1.341830e-15 20
21 1.341830e-15 21
22 1.341831e-15 22
23 1.341831e-15 23
24 1.341832e-15 24
25 1.341833e-15 25
26 1.341833e-15 26
27 1.341834e-15 27
28 1.341835e-15 28
29 1.341835e-15 29
30 1.341836e-15 30
```

From the above table we can see that the likelihood asymptotes in $\hat{\beta} + s\eta$ as $s \to \infty$.

part d

```
b = c(0,1,0,0)
 library(data.table)
 foo = setDT(as.data.frame(cbind(m$y, m$x %*% b)))
 colnames(foo) = c("y", "sep")
 foo[, .(.N), by = c("y", "sep")]
```

```
   y sep  N
1: 0   0 49
2: 1   0 17
3: 1   1 13
```

We can see that there is no (y,sep) which is (0,1) in the data set. We observe quasi-complete separation in NV. If we proceed with a part of the data after removing the columns corresponding to NV=1, then we won't get an estimate of the coefficient corresponding to NV (which is NA in the summary of 'glmdr'). But in that case, we can see that the eigen vector, η will be (almost) zero. So, for any scalar s, $s\eta$ will be (almost) zero, and hence, $\hat{\beta} + s\eta$ will not change as $s \to \infty$. Therefore, the likelihood asymptotes in $\hat{\beta} + s\eta$ as $s \to \infty$.

Problem 6 [10 points]: Summarise the Firth approach mentioned in Section 7.4.7 and 7.4.8 of Agresti. Compare and contrast the Firth approach with the direct MLE approach outlined in the complete separation notes. What are the strengths and weaknesses of each approach?

Answer:

Firth approach

David Firth showed that the ML estimator in logistic regression is biased away from 0 and proposed a penalized likelihood correction that reduces the bias. Maximizing the Firth penalized log-likelihood function yields an estimate that always exists and is unique. The penalized likelihood estimates are often more believable than ML estimates, such as when ML estimates are infinite or badly affected by multi-collinearity.

Firth's Penalized Likelihood for Logistic Regression

For most models the ML estimator $\hat{\boldsymbol{\beta}}$ has bias on the order of $1/n$, and Firth showed how to penalize the log likelihood such that this reduces to order $1/n^2$. For the canonical parameter of an exponential family model, the penalized log-likelihood function utilizes the determinant of the information matrix $\mathcal{J}$,

$$L^*(\boldsymbol{\beta}) = L(\boldsymbol{\beta}) + \frac{1}{2}\log|\mathcal{J}|.$$

For application to logistic regression, Firth noted that when the model matrix is of full rank, $\log|\mathcal{J}|$ is strictly concave. Maximizing the penalized likelihood yields a maximum penalized likelihood estimate that always exists and is unique. This penalized likelihood then is proportional to the Bayesian posterior distribution resulting from using the Jeffreys prior.

One situation in which Firth's penalized likelihood estimate is very helpful is when complete or quasi-complete separation occurs in the space of explanatory variables. Then, ordinary ML estimates of logistic regression parameters are infinite or do not exist, but the penalized estimator is finite.

** Compare and contrast the Firth approach with the direct MLE approach**

- The main objective of the Firth approach is to address the problem of bias in maximum likelihood estimates caused by separation, which occurs when one or more predictor variables perfectly predict the outcome variable. Whereas, the direct MLE approach aims to maximize the likelihood function directly to estimate the parameters of the model without adjusting for separation.
- Firth's approach incorporates a penalized likelihood adjustment that adds a small bias to the log-likelihood function to counteract the separation issue. On the other hand, the direct MLE approach does not incorporate any adjustments for separation.
- Firth approach typically involves additional computational steps compared to the direct MLE approach because of the penalized likelihood adjustment. However, the direct MLE approach is usually computationally simpler compared to the Firth approach since it doesn't involve any additional adjustments.
- By adding a small bias to the log-likelihood function, the Firth approach corrects for the bias introduced by separation, resulting in less biased parameter estimates. However, the direct MLE approach may produce biased parameter estimates when dealing with separated data.

Problem 7 [10 points]: Use `glmdr` software to analyze the `catrec.txt` data using Poisson regression. Specifically, fit a third order model and provide confidence intervals for all mean-value parameter estimates, both one-sided intervals for responses that are constrained on the boundary and two-sided intervals for responses that are unconstrained. Also verify that the third order model is appropriate using a likelihood ratio test.

Answer:

```r
library(nloptr)
library(mdscore)
#Loading the data
data=read.table('/Users/diptarka/Documents/GitHub/STAT 528 Diptarka/catrec.txt',header = TRUE)
head(data)
```

```
  v1 v2 v3 v4 v5 v6 v7 y
1  0  0  0  0  0  0  0 0
2  1  0  0  0  0  0  0 8
3  0  1  0  0  0  0  0 7
4  1  1  0  0  0  0  0 8
5  0  0  1  0  0  0  0 9
6  1  0  1  0  0  0  0 7
```

```r
#Using glmdr software to fit the 3rd order Poisson regression model
mod = glmdr(y ~ .^3,family="poisson",data)
summary(mod)
```

```
MLE exists in Barndorff-Nielsen completion
it is conditional on components of the response
corresponding to object$linearity == FALSE being
conditioned on their observed values




GLM summary for limiting conditional model


Call:
stats::glm(formula = y ~ .^3, family = "poisson", data = data,
    subset = c("2", "3", "4", "5", "6", "7", "8", "10", "11",
    "12", "13", "14", "15", "16", "17", "18", "19", "21", "22",
    "23", "24", "25", "26", "27", "29", "30", "31", "32", "34",
    "35", "36", "37", "38", "39", "40", "42", "43", "44", "45",
    "46", "47", "48", "49", "50", "51", "53", "54", "55", "56",
    "57", "58", "59", "61", "62", "63", "64", "66", "67", "68",
    "69", "70", "71", "72", "74", "75", "76", "77", "78", "79",
    "80", "81", "82", "83", "85", "86", "87", "88", "89", "90",
    "91", "93", "94", "95", "96", "98", "99", "100", "101", "102",
    "103", "104", "106", "107", "108", "109", "110", "111", "112",
    "113", "114", "115", "117", "118", "119", "120", "121", "122",
    "123", "125", "126", "127", "128"), x = TRUE, y = TRUE)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.63571  -0.30009  -0.02353   0.27258   1.42540

Coefficients: (1 not defined because of singularities)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.150481   0.585423   3.673 0.000239 ***
v1           0.069795   0.587067   0.119 0.905364
v2          -0.524215   0.513583  -1.021 0.307396
```

```
v3           0.052966   0.551965   0.096 0.923552
v4          -0.709525   0.580147  -1.223 0.221326
v5           0.243002   0.548686   0.443 0.657853
v6          -1.163256   0.563668  -2.064 0.039044 *
v7          -0.990704   0.597335  -1.659 0.097208 .
v1:v2        0.384345   0.543024   0.708 0.479079
v1:v3       -0.630375   0.570151  -1.106 0.268888
v1:v4        0.008801   0.511458   0.017 0.986271
v1:v5       -1.022805   0.570440  -1.793 0.072971 .
v1:v6        0.540164   0.493879   1.094 0.274079
v1:v7        0.097178   0.536628   0.181 0.856297
v2:v3        0.602411   0.437371   1.377 0.168405
v2:v4        0.748226   0.486811   1.537 0.124295
v2:v5       -0.068926   0.428100  -0.161 0.872090
v2:v6        0.297165   0.487409   0.610 0.542071
v2:v7        0.274198   0.508369   0.539 0.589634
v3:v4       -0.124465   0.541056  -0.230 0.818060
v3:v5       -0.439354   0.468418  -0.938 0.348268
v3:v6        0.024399   0.530220   0.046 0.963296
v3:v7       -0.104400   0.556960  -0.187 0.851310
v4:v5       -0.169421   0.521323  -0.325 0.745194
v4:v6        0.756513   0.474213   1.595 0.110644
v4:v7        0.780671   0.500911   1.559 0.119114
v5:v6        1.245629   0.510770   2.439 0.014739 *
v5:v7       -0.262620   0.523125  -0.502 0.615652
v6:v7        0.697014   0.489957   1.423 0.154852
v1:v2:v3    -0.349902   0.483330  -0.724 0.469102
v1:v2:v4     0.101569   0.389778   0.261 0.794416
v1:v2:v5     0.655208   0.493737   1.327 0.184496
v1:v2:v6    -0.329286   0.390979  -0.842 0.399670
v1:v2:v7    -0.520368   0.393042  -1.324 0.185520
v1:v3:v4     0.353292   0.406623   0.869 0.384932
v1:v3:v5     0.638711   0.484979   1.317 0.187843
v1:v3:v6     0.352694   0.402715   0.876 0.381143
v1:v3:v7    -0.001586   0.413554  -0.004 0.996941
v1:v4:v5     0.664745   0.400212   1.661 0.096717 .
v1:v4:v6    -0.463885   0.368214  -1.260 0.207732
v1:v4:v7    -0.342583   0.372009  -0.921 0.357103
v1:v5:v6     0.044968   0.399958   0.112 0.910481
v1:v5:v7     0.447641   0.404364   1.107 0.268283
v1:v6:v7     0.218868   0.371499   0.589 0.555763
v2:v3:v4    -0.325914   0.404392  -0.806 0.420280
v2:v3:v5           NA         NA      NA       NA
v2:v3:v6    -0.247853   0.405621  -0.611 0.541168
v2:v3:v7     0.028322   0.414520   0.068 0.945527
v2:v4:v5     0.004655   0.394418   0.012 0.990583
v2:v4:v6    -0.111152   0.373713  -0.297 0.766141
v2:v4:v7    -0.148061   0.376692  -0.393 0.694279
v2:v5:v6    -0.766051   0.394925  -1.940 0.052412 .
v2:v5:v7     0.075213   0.399004   0.189 0.850482
v2:v6:v7     0.460826   0.381109   1.209 0.226597
v3:v4:v5    -0.063494   0.423318  -0.150 0.880771
v3:v4:v6     0.357746   0.366298   0.977 0.328741
v3:v4:v7    -0.106368   0.371567  -0.286 0.774672
```

```
v3:v5:v6    -0.234816   0.422424  -0.556 0.578295
v3:v5:v7     0.804923   0.423843   1.899 0.057550 .
v3:v6:v7    -0.659090   0.371085  -1.776 0.075714 .
v4:v5:v6    -0.427957   0.375755  -1.139 0.254734
v4:v5:v7     0.125167   0.377356   0.332 0.740119
v4:v6:v7     0.014192   0.370131   0.038 0.969413
v5:v6:v7    -0.811516   0.377098  -2.152 0.031397 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 156.215  on 111  degrees of freedom
Residual deviance:  31.291  on  49  degrees of freedom
AIC: 526.46

Number of Fisher Scoring iterations: 5
```

```
# One-sided Confidence Intervals
CIs = inference(mod)
CIs
```

```
   lower      upper
1      0 0.28630975
2      0 0.14082947
3      0 0.21996698
4      0 0.42095569
5      0 0.08946242
6      0 0.09376644
7      0 0.19302341
8      0 0.28869769
9      0 0.10631113
10     0 0.11415033
11     0 0.09128766
12     0 0.26461097
13     0 0.06669488
14     0 0.15477613
15     0 0.14096916
16     0 0.32392015
```

In above we have used the inference function to obtain one-sided confidence intervals for mean-value parameters corresponding to components that are constrained on the boundary. Now we shall show the two sided intervals for responses that are unconstrained.

```
mod1=glm(y ~ .^3,family="poisson",data=data, x=TRUE,y=TRUE)
mod2=update(mod1,subset=mod$linearity)
summary(mod2)
```

```
Call:
glm(formula = y ~ .^3, family = "poisson", data = data, subset = mod$linearity,
    x = TRUE, y = TRUE)
```

```
Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.63571  -0.30009  -0.02353   0.27258   1.42540

Coefficients: (1 not defined because of singularities)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.150481   0.585423   3.673 0.000239 ***
v1           0.069795   0.587067   0.119 0.905364
v2          -0.524215   0.513583  -1.021 0.307396
v3           0.052966   0.551965   0.096 0.923552
v4          -0.709525   0.580147  -1.223 0.221326
v5           0.243002   0.548686   0.443 0.657853
v6          -1.163256   0.563668  -2.064 0.039044 *
v7          -0.990704   0.597335  -1.659 0.097208 .
v1:v2        0.384345   0.543024   0.708 0.479079
v1:v3       -0.630375   0.570151  -1.106 0.268888
v1:v4        0.008801   0.511458   0.017 0.986271
v1:v5       -1.022805   0.570440  -1.793 0.072971 .
v1:v6        0.540164   0.493879   1.094 0.274079
v1:v7        0.097178   0.536628   0.181 0.856297
v2:v3        0.602411   0.437371   1.377 0.168405
v2:v4        0.748226   0.486811   1.537 0.124295
v2:v5       -0.068926   0.428100  -0.161 0.872090
v2:v6        0.297165   0.487409   0.610 0.542071
v2:v7        0.274198   0.508369   0.539 0.589634
v3:v4       -0.124465   0.541056  -0.230 0.818060
v3:v5       -0.439354   0.468418  -0.938 0.348268
v3:v6        0.024399   0.530220   0.046 0.963296
v3:v7       -0.104400   0.556960  -0.187 0.851310
v4:v5       -0.169421   0.521323  -0.325 0.745194
v4:v6        0.756513   0.474213   1.595 0.110644
v4:v7        0.780671   0.500911   1.559 0.119114
v5:v6        1.245629   0.510770   2.439 0.014739 *
v5:v7       -0.262620   0.523125  -0.502 0.615652
v6:v7        0.697014   0.489957   1.423 0.154852
v1:v2:v3    -0.349902   0.483330  -0.724 0.469102
v1:v2:v4     0.101569   0.389778   0.261 0.794416
v1:v2:v5     0.655208   0.493737   1.327 0.184496
v1:v2:v6    -0.329286   0.390979  -0.842 0.399670
v1:v2:v7    -0.520368   0.393042  -1.324 0.185520
v1:v3:v4     0.353292   0.406623   0.869 0.384932
v1:v3:v5     0.638711   0.484979   1.317 0.187843
v1:v3:v6     0.352694   0.402715   0.876 0.381143
v1:v3:v7    -0.001586   0.413554  -0.004 0.996941
v1:v4:v5     0.664745   0.400212   1.661 0.096717 .
v1:v4:v6    -0.463885   0.368214  -1.260 0.207732
v1:v4:v7    -0.342583   0.372009  -0.921 0.357103
v1:v5:v6     0.044968   0.399958   0.112 0.910481
v1:v5:v7     0.447641   0.404364   1.107 0.268283
v1:v6:v7     0.218868   0.371499   0.589 0.555763
v2:v3:v4    -0.325914   0.404392  -0.806 0.420280
v2:v3:v5           NA         NA      NA       NA
v2:v3:v6    -0.247853   0.405621  -0.611 0.541168
v2:v3:v7     0.028322   0.414520   0.068 0.945527
```

```
v2:v4:v5     0.004655   0.394418   0.012 0.990583
v2:v4:v6    -0.111152   0.373713  -0.297 0.766141
v2:v4:v7    -0.148061   0.376692  -0.393 0.694279
v2:v5:v6    -0.766051   0.394925  -1.940 0.052412 .
v2:v5:v7     0.075213   0.399004   0.189 0.850482
v2:v6:v7     0.460826   0.381109   1.209 0.226597
v3:v4:v5    -0.063494   0.423318  -0.150 0.880771
v3:v4:v6     0.357746   0.366298   0.977 0.328741
v3:v4:v7    -0.106368   0.371567  -0.286 0.774672
v3:v5:v6    -0.234816   0.422424  -0.556 0.578295
v3:v5:v7     0.804923   0.423843   1.899 0.057550 .
v3:v6:v7    -0.659090   0.371085  -1.776 0.075714 .
v4:v5:v6    -0.427957   0.375755  -1.139 0.254734
v4:v5:v7     0.125167   0.377356   0.332 0.740119
v4:v6:v7     0.014192   0.370131   0.038 0.969413
v5:v6:v7    -0.811516   0.377098  -2.152 0.031397 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 156.215  on 111  degrees of freedom
Residual deviance:  31.291  on  49  degrees of freedom
AIC: 526.46

Number of Fisher Scoring iterations: 5
```

```r
pred=predict(mod2,se.fit = TRUE, type = "response")

CIs_two=cbind(pred$fit-qnorm(0.975)*(pred$se.fit),
              pred$fit+qnorm(0.975)*(pred$se.fit))
CIs_two
```

```
          [,1]      [,2]
2    4.1225794 14.297167
3    1.6553214  8.514389
4    3.3515950 12.663846
5    3.8973641 14.215000
6    1.8159636  8.523987
7    4.7546085 14.830931
8    2.2138503  9.358914
10   1.3002430  7.840116
11   1.7609413  8.810062
12   4.1471451 14.442860
13   0.8782774  6.988161
14   0.7813852  5.668862
15   2.6610345 10.315125
16   2.3788529  9.811419
17   5.3492799 16.553863
18   1.2217860  7.223596
19   2.1622088  9.941231
21   3.2718090 11.611529
22   0.7955744  4.991161
23   3.3100229 11.711938
```

24	1.9928071	9.647679
25	1.3337452	7.760888
26	0.7610721	6.116189
27	1.7134991	8.956378
29	0.5935347	4.527403
30	0.6511110	4.908353
31	1.2844978	6.636959
32	4.2062794	14.765606
34	1.5971712	8.281046
35	0.3040465	3.973260
36	1.1712670	7.146063
37	0.4357698	5.363538
38	1.2017812	6.883276
39	0.8234526	5.764531
40	0.9158568	5.923424
42	0.7981173	5.770057
43	1.2119806	7.265265
44	2.1133668	9.462096
45	0.8073299	6.867973
46	1.5877670	8.077067
47	2.2379602	9.663679
48	2.4495287	9.903330
49	5.9446631	17.839107
50	3.5085804	12.954586
51	1.0403033	7.183045
53	2.6027717	10.491818
54	2.6195220	10.385711
55	0.8175962	5.636603
56	1.2767174	7.914492
57	2.6313072	11.085386
58	2.1088480	9.600608
59	1.2835907	7.726240
61	1.4178212	7.531662
62	3.4004391	12.206277
63	0.7909976	5.258302
64	3.5400126	13.202993
66	0.8538205	6.683730
67	0.3978836	4.569568
68	0.4086172	4.714980
69	0.4249484	5.633720
70	0.2032951	3.602417
71	1.3896357	7.476210
72	0.2116017	3.214057
74	0.4991003	5.297443
75	1.5228062	8.197511
76	1.0893371	6.858939
77	0.2472298	4.917246
78	0.1508936	3.157049
79	1.7689727	8.173054
80	0.4423221	3.893612
81	0.6371389	5.617391
82	0.2271642	3.931172
83	0.3583352	4.543338
85	1.2940929	7.268820

```
86   0.6034011  5.128230
87   2.4191916 10.190232
88   1.4709159  8.526093
89   0.6437610  5.781252
90   0.5280266  5.419476
91   1.3359634  7.883246
93   0.8062256  5.747319
94   1.2744985  7.421137
95   2.5465551 10.208352
96   5.3111995 16.877472
98   1.5297011  8.571835
99   0.6635226  5.986745
100  1.6346324  8.907213
101 -0.1427022  2.157578
102  0.2214164  3.625596
103  0.3957688  4.514910
104  0.2985760  3.850724
106  1.7305100  8.827363
107  6.3944030 18.772296
108  4.6806200 15.207423
109  0.2597288  5.048838
110  0.8071847  5.692058
111  3.2387386 11.991389
112  1.5722143  7.561718
113  0.6057880  5.450983
114  1.2924829  7.705369
115  0.2796504  4.428939
117  0.2232701  3.252460
118  1.0149417  6.381940
119  0.2857476  3.676834
120  0.7615243  6.424554
121  1.2532205  7.512227
122  2.0453060  9.355738
123  1.8875546  9.276477
125  0.6060110  4.753452
126  2.9814260 11.236278
127  1.0949753  6.131196
128  4.0175398 14.064114
```

Now we shall verify whether the third order model is appropriate using a likelihood ratio test.

```
mod.simple=glmdr(y ~ .,family="poisson",data)
mod1.simple=glm(y ~ .,family="poisson",data=data, x=TRUE,y=TRUE)
mod2.simple=update(mod1.simple,subset=mod.simple$linearity)

mod.2=glmdr(y ~ .^2,family="poisson",data)
mod1.2=glm(y ~ .^2,family="poisson",data=data, x=TRUE,y=TRUE)
mod2.2=update(mod1.2,subset=mod.2$linearity)

mod.4=glmdr(y ~ .^4,family="poisson",data)
mod1.4=glm(y ~ .^4,family="poisson",data=data, x=TRUE,y=TRUE)
mod2.4=update(mod1.4,subset=mod.4$linearity)

lr.test(mod2.2,mod2)
```

```
$LR
[1] 160.3381

$pvalue
[1] 1.729207e-13

attr(,"class")
[1] "lrt.test"
```

```
lr.test(mod2.simple,mod2)
```

```
$LR
[1] 246.5368

$pvalue
[1] 3.160119e-21

attr(,"class")
[1] "lrt.test"
```

```
lr.test(mod2.simple,mod2.2)
```

```
$LR
[1] 86.19868

$pvalue
[1] 7.245628e-10

attr(,"class")
[1] "lrt.test"
```

```
lr.test(mod2,mod2.4)
```

```
$LR
[1] 15.22438

$pvalue
[1] 0.9921224

attr(,"class")
[1] "lrt.test"
```

From the above results of likelihood ratio tests, we can see that when we compare our model with 3rd order terms, 'mod2' with other models with lower orders ('mod2.simple', 'mod2.2'), then the p-values are very low (almost zero). So, we can say that the 3rd order terms are significant. Moreover, the likelihood ratio test between 'mod2' and the model with 4th order terms ('mod2.4') shows that p-value is not significant (LR follows chi-square). So, the 4th order terms are not necessary in this model. Therefore, we can conclude that the third order model is appropriate.

In above we have used the inference function to obtain one-sided confidence intervals for mean-value parameters corresponding to components that are constrained on the boundary. Now we shall show the two sided intervals for responses that are unconstrained.

```r
mod1=glm(y ~ .^3,family="poisson",data=data, x=TRUE,y=TRUE)
mod2=update(mod1,subset=mod$linearity)
summary(mod2)
```

```
Call:
glm(formula = y ~ .^3, family = "poisson", data = data, subset = mod$linearity,
    x = TRUE, y = TRUE)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.63571  -0.30009  -0.02353   0.27258   1.42540

Coefficients: (1 not defined because of singularities)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.150481   0.585423   3.673 0.000239 ***
v1           0.069795   0.587067   0.119 0.905364
v2          -0.524215   0.513583  -1.021 0.307396
v3           0.052966   0.551965   0.096 0.923552
v4          -0.709525   0.580147  -1.223 0.221326
v5           0.243002   0.548686   0.443 0.657853
v6          -1.163256   0.563668  -2.064 0.039044 *
v7          -0.990704   0.597335  -1.659 0.097208 .
v1:v2        0.384345   0.543024   0.708 0.479079
v1:v3       -0.630375   0.570151  -1.106 0.268888
v1:v4        0.008801   0.511458   0.017 0.986271
v1:v5       -1.022805   0.570440  -1.793 0.072971 .
v1:v6        0.540164   0.493879   1.094 0.274079
v1:v7        0.097178   0.536628   0.181 0.856297
v2:v3        0.602411   0.437371   1.377 0.168405
v2:v4        0.748226   0.486811   1.537 0.124295
v2:v5       -0.068926   0.428100  -0.161 0.872090
v2:v6        0.297165   0.487409   0.610 0.542071
v2:v7        0.274198   0.508369   0.539 0.589634
v3:v4       -0.124465   0.541056  -0.230 0.818060
v3:v5       -0.439354   0.468418  -0.938 0.348268
v3:v6        0.024399   0.530220   0.046 0.963296
v3:v7       -0.104400   0.556960  -0.187 0.851310
v4:v5       -0.169421   0.521323  -0.325 0.745194
v4:v6        0.756513   0.474213   1.595 0.110644
v4:v7        0.780671   0.500911   1.559 0.119114
v5:v6        1.245629   0.510770   2.439 0.014739 *
v5:v7       -0.262620   0.523125  -0.502 0.615652
v6:v7        0.697014   0.489957   1.423 0.154852
v1:v2:v3    -0.349902   0.483330  -0.724 0.469102
v1:v2:v4     0.101569   0.389778   0.261 0.794416
v1:v2:v5     0.655208   0.493737   1.327 0.184496
v1:v2:v6    -0.329286   0.390979  -0.842 0.399670
v1:v2:v7    -0.520368   0.393042  -1.324 0.185520
v1:v3:v4     0.353292   0.406623   0.869 0.384932
v1:v3:v5     0.638711   0.484979   1.317 0.187843
v1:v3:v6     0.352694   0.402715   0.876 0.381143
v1:v3:v7    -0.001586   0.413554  -0.004 0.996941
```

```
v1:v4:v5     0.664745   0.400212   1.661 0.096717 .
v1:v4:v6    -0.463885   0.368214  -1.260 0.207732
v1:v4:v7    -0.342583   0.372009  -0.921 0.357103
v1:v5:v6     0.044968   0.399958   0.112 0.910481
v1:v5:v7     0.447641   0.404364   1.107 0.268283
v1:v6:v7     0.218868   0.371499   0.589 0.555763
v2:v3:v4    -0.325914   0.404392  -0.806 0.420280
v2:v3:v5           NA         NA      NA       NA
v2:v3:v6    -0.247853   0.405621  -0.611 0.541168
v2:v3:v7     0.028322   0.414520   0.068 0.945527
v2:v4:v5     0.004655   0.394418   0.012 0.990583
v2:v4:v6    -0.111152   0.373713  -0.297 0.766141
v2:v4:v7    -0.148061   0.376692  -0.393 0.694279
v2:v5:v6    -0.766051   0.394925  -1.940 0.052412 .
v2:v5:v7     0.075213   0.399004   0.189 0.850482
v2:v6:v7     0.460826   0.381109   1.209 0.226597
v3:v4:v5    -0.063494   0.423318  -0.150 0.880771
v3:v4:v6     0.357746   0.366298   0.977 0.328741
v3:v4:v7    -0.106368   0.371567  -0.286 0.774672
v3:v5:v6    -0.234816   0.422424  -0.556 0.578295
v3:v5:v7     0.804923   0.423843   1.899 0.057550 .
v3:v6:v7    -0.659090   0.371085  -1.776 0.075714 .
v4:v5:v6    -0.427957   0.375755  -1.139 0.254734
v4:v5:v7     0.125167   0.377356   0.332 0.740119
v4:v6:v7     0.014192   0.370131   0.038 0.969413
v5:v6:v7    -0.811516   0.377098  -2.152 0.031397 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 156.215  on 111  degrees of freedom
Residual deviance:  31.291  on  49  degrees of freedom
AIC: 526.46

Number of Fisher Scoring iterations: 5
```

```r
pred=predict(mod2,se.fit = TRUE, type = "response")

CIs_two=cbind(pred$fit-qnorm(0.975)*(pred$se.fit),
              pred$fit+qnorm(0.975)*(pred$se.fit))
CIs_two
```

```
          [,1]      [,2]
2    4.1225794 14.297167
3    1.6553214  8.514389
4    3.3515950 12.663846
5    3.8973641 14.215000
6    1.8159636  8.523987
7    4.7546085 14.830931
8    2.2138503  9.358914
10   1.3002430  7.840116
11   1.7609413  8.810062
12   4.1471451 14.442860
```

13	0.8782774	6.988161
14	0.7813852	5.668862
15	2.6610345	10.315125
16	2.3788529	9.811419
17	5.3492799	16.553863
18	1.2217860	7.223596
19	2.1622088	9.941231
21	3.2718090	11.611529
22	0.7955744	4.991161
23	3.3100229	11.711938
24	1.9928071	9.647679
25	1.3337452	7.760888
26	0.7610721	6.116189
27	1.7134991	8.956378
29	0.5935347	4.527403
30	0.6511110	4.908353
31	1.2844978	6.636959
32	4.2062794	14.765606
34	1.5971712	8.281046
35	0.3040465	3.973260
36	1.1712670	7.146063
37	0.4357698	5.363538
38	1.2017812	6.883276
39	0.8234526	5.764531
40	0.9158568	5.923424
42	0.7981173	5.770057
43	1.2119806	7.265265
44	2.1133668	9.462096
45	0.8073299	6.867973
46	1.5877670	8.077067
47	2.2379602	9.663679
48	2.4495287	9.903330
49	5.9446631	17.839107
50	3.5085804	12.954586
51	1.0403033	7.183045
53	2.6027717	10.491818
54	2.6195220	10.385711
55	0.8175962	5.636603
56	1.2767174	7.914492
57	2.6313072	11.085386
58	2.1088480	9.600608
59	1.2835907	7.726240
61	1.4178212	7.531662
62	3.4004391	12.206277
63	0.7909976	5.258302
64	3.5400126	13.202993
66	0.8538205	6.683730
67	0.3978836	4.569568
68	0.4086172	4.714980
69	0.4249484	5.633720
70	0.2032951	3.602417
71	1.3896357	7.476210
72	0.2116017	3.214057
74	0.4991003	5.297443

```
75   1.5228062  8.197511
76   1.0893371  6.858939
77   0.2472298  4.917246
78   0.1508936  3.157049
79   1.7689727  8.173054
80   0.4423221  3.893612
81   0.6371389  5.617391
82   0.2271642  3.931172
83   0.3583352  4.543338
85   1.2940929  7.268820
86   0.6034011  5.128230
87   2.4191916 10.190232
88   1.4709159  8.526093
89   0.6437610  5.781252
90   0.5280266  5.419476
91   1.3359634  7.883246
93   0.8062256  5.747319
94   1.2744985  7.421137
95   2.5465551 10.208352
96   5.3111995 16.877472
98   1.5297011  8.571835
99   0.6635226  5.986745
100  1.6346324  8.907213
101 -0.1427022  2.157578
102  0.2214164  3.625596
103  0.3957688  4.514910
104  0.2985760  3.850724
106  1.7305100  8.827363
107  6.3944030 18.772296
108  4.6806200 15.207423
109  0.2597288  5.048838
110  0.8071847  5.692058
111  3.2387386 11.991389
112  1.5722143  7.561718
113  0.6057880  5.450983
114  1.2924829  7.705369
115  0.2796504  4.428939
117  0.2232701  3.252460
118  1.0149417  6.381940
119  0.2857476  3.676834
120  0.7615243  6.424554
121  1.2532205  7.512227
122  2.0453060  9.355738
123  1.8875546  9.276477
125  0.6060110  4.753452
126  2.9814260 11.236278
127  1.0949753  6.131196
128  4.0175398 14.064114
```

Now we shall verify whether the third order model is appropriate using a likelihood ratio test.

```
mod.simple=glmdr(y ~ .,family="poisson",data)
mod1.simple=glm(y ~ .,family="poisson",data=data, x=TRUE,y=TRUE)
mod2.simple=update(mod1.simple,subset=mod.simple$linearity)
```

```r
mod.2=glmdr(y ~ .^2,family="poisson",data)
mod1.2=glm(y ~ .^2,family="poisson",data=data, x=TRUE,y=TRUE)
mod2.2=update(mod1.2,subset=mod.2$linearity)

mod.4=glmdr(y ~ .^4,family="poisson",data)
mod1.4=glm(y ~ .^4,family="poisson",data=data, x=TRUE,y=TRUE)
mod2.4=update(mod1.4,subset=mod.4$linearity)

lr.test(mod2.2,mod2)
```

```
$LR
[1] 160.3381

$pvalue
[1] 1.729207e-13

attr(,"class")
[1] "lrt.test"
```

```r
lr.test(mod2.simple,mod2)
```

```
$LR
[1] 246.5368

$pvalue
[1] 3.160119e-21

attr(,"class")
[1] "lrt.test"
```

```r
lr.test(mod2.simple,mod2.2)
```

```
$LR
[1] 86.19868

$pvalue
[1] 7.245628e-10

attr(,"class")
[1] "lrt.test"
```

```r
lr.test(mod2,mod2.4)
```

```
$LR
[1] 15.22438

$pvalue
[1] 0.9921224

attr(,"class")
[1] "lrt.test"
```

From the above results of likelihood ratio tests, we can see that when we compare our model with 3rd order terms, 'mod2' with other models with lower orders ('mod2.simple', 'mod2.2'), then the p-values are very low (almost zero). So, we can say that the 3rd order terms are significant. Moreover, the likelihood ratio test between 'mod2' and the model with 4th order terms ('mod2.4') shows that p-value is not significant (LR follows chi-square). So, the 4th order terms are not necessary in this model. Therefore, we can conclude that the third order model is appropriate.