Homework 4

MacKenzie Ullman

3/15/2021

```r
#setwd("F:/Quant Eco/")
#getwd()
hw4 <- read.csv(file= 'Homework 4 Data.csv')
head(hw4)
```

```
##    y         x1 x2
## 1  4 -2.4335748  a
## 2  3 -0.6850696  b
## 3  5 -0.8038049  a
## 4  5  2.1243703  b
## 5  2 -0.3157032  b
## 6 10  0.1981158  a
```

```r
summary(hw4)
```

```
##        y               x1                x2
##  Min.   : 0.00   Min.   :-2.4336   Length:100
##  1st Qu.: 2.00   1st Qu.:-0.6994   Class :character
##  Median : 4.00   Median :-0.1466   Mode  :character
##  Mean   : 4.48   Mean   :-0.1777
##  3rd Qu.: 6.25   3rd Qu.: 0.3521
##  Max.   :14.00   Max.   : 2.1244
```

Fit a Poisson regression model that assumes expected count is an interactive function of variables x1 and x2.

```r
fit <- glm( y ~ x1 * x2, family= poisson, data= hw4)
summary(fit)
```

```
##
## Call:
## glm(formula = y ~ x1 * x2, family = poisson, data = hw4)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1523  -0.6131  -0.1399   0.4250   2.5643
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.85710    0.05822  31.896  < 2e-16 ***
## x1          -0.09937    0.06353  -1.564 0.117778
## x2b         -1.04662    0.11283  -9.276  < 2e-16 ***
## x1:x2b       0.47840    0.12314   3.885 0.000102 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 208.324  on 99  degrees of freedom
## Residual deviance:  84.732  on 96  degrees of freedom
## AIC: 405.67
##
## Number of Fisher Scoring iterations: 4
```

Interpret the effect of variable x1 on the expected count when x2 is fixed at level "b". Verify your interpretation in R.

```
beta <- coef(fit)
beta
```

```
## (Intercept)          x1         x2b      x1:x2b
##  1.85710253 -0.09937274 -1.04662374  0.47839525
```

```
x2 <- 1
c_1<- beta[2] + beta[4] * 1
c_1
```

```
##        x1
## 0.3790225
```

```
exp(c_1)
```

```
##       x1
## 1.460856
```

```
# Lambda increases by 46% for every 1 unit increase in x1
```

```
#verify
c_2 <- beta[2] + beta[4] * 2
c_2
```

```
##        x1
## 0.8574178
```

```
exp(c_2)
```

```
##       x1
## 2.357066
```

```
exp(c_2) / exp(c_1)
```

```
##       x1
## 1.613483
```

```
exp(beta[4])
```

```
##   x1:x2b
## 1.613483
```

Interpret the effect of variable x2 on the expected count when x1 is fixed at 1. Verify your interpretation in R.

```
beta
```

```
## (Intercept)          x1         x2b      x1:x2b
##  1.85710253 -0.09937274 -1.04662374  0.47839525
```

```
x1 <- 1
c_x2 <- beta[3] + beta[4] * 1
c_x2
```

```
##        x2b
## -0.5682285
```

```
exp(c_x2)
```

```
##       x2b
## 0.5665282
```

```
# Lambda increases by 56% for every 1 unit increase in x2
```

```
#verify
c2_x2 <- beta[3] + beta[4] * 2
c2_x2
```

```
##         x2b
## -0.08983324
```

```
exp(c2_x2)
```

```
##       x2b
## 0.9140836
```

```
exp(c2_x2)/ exp(c_x2)
```

```
##      x2b
## 1.613483
```

```
exp(beta[4])
```

```
##   x1:x2b
## 1.613483
```

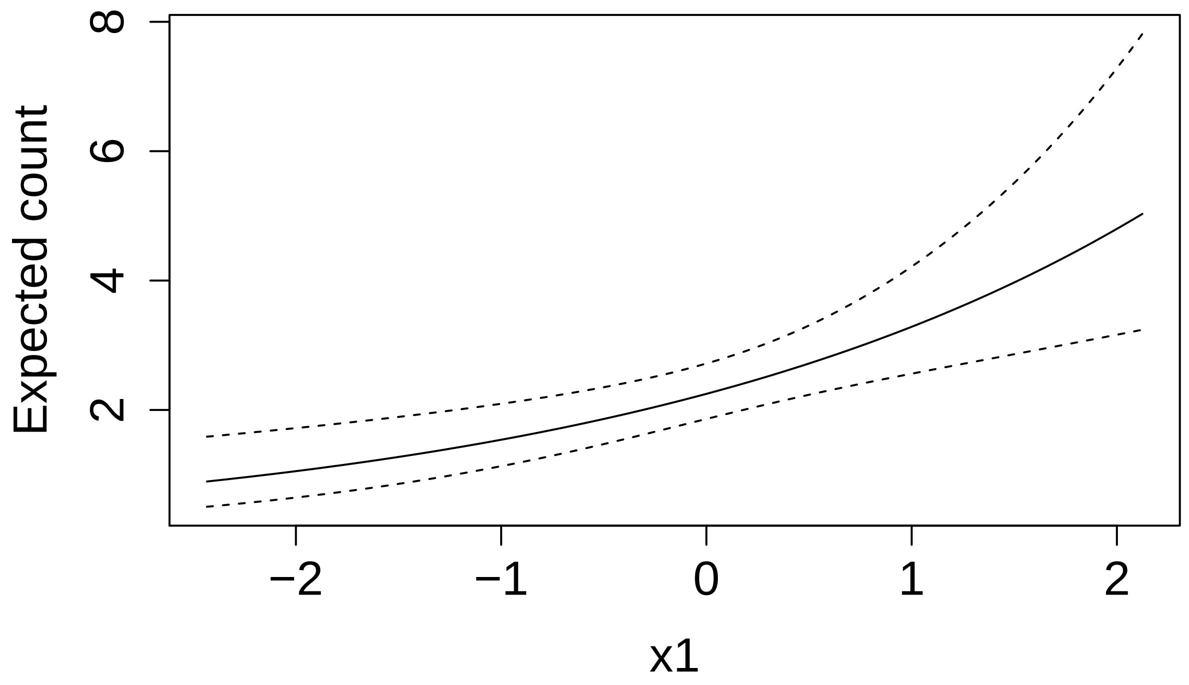
Predict the expected count, ± 95% confidence intervals, over the observed range of values of x1, assuming x2 is fixed at level "b".

```
# x1 range of values
x1_new <- data.frame(
  x1 = seq(min(hw4$x1), max(hw4$x1), length.out = 100),
  x2 = factor(rep('b', times= 100), levels = c('a','b')))

# predicted expected count
prd <- predict.glm(object= fit, newdata = x1_new, type = 'link', se.fit = T)
prd
```

```
## $fit
##            1            2            3            4            5            6 
## -0.111900830 -0.094450691 -0.077000552 -0.059550413 -0.042100273 -0.024650134 
##            7            8            9           10           11           12 
## -0.007199995  0.010250144  0.027700283  0.045150422  0.062600562  0.080050701 
##           13           14           15           16           17           18 
##  0.097500840  0.114950979  0.132401118  0.149851257  0.167301397  0.184751536 
##           19           20           21           22           23           24 
##  0.202201675  0.219651814  0.237101953  0.254552093  0.272002232  0.289452371 
##           25           26           27           28           29           30 
##  0.306902510  0.324352649  0.341802788  0.359252928  0.376703067  0.394153206 
##           31           32           33           34           35           36 
##  0.411603345  0.429053484  0.446503623  0.463953763  0.481403902  0.498854041 
##           37           38           39           40           41           42 
##  0.516304180  0.533754319  0.551204459  0.568654598  0.586104737  0.603554876 
##           43           44           45           46           47           48 
##  0.621005015  0.638455154  0.655905294  0.673355433  0.690805572  0.708255711 
##           49           50           51           52           53           54 
##  0.725705850  0.743155990  0.760606129  0.778056268  0.795506407  0.812956546 
##           55           56           57           58           59           60 
##  0.830406685  0.847856825  0.865306964  0.882757103  0.900207242  0.917657381 
##           61           62           63           64           65           66 
##  0.935107520  0.952557660  0.970007799  0.987457938  1.004908077  1.022358216 
##           67           68           69           70           71           72 
##  1.039808356  1.057258495  1.074708634  1.092158773  1.109608912  1.127059051 
##           73           74           75           76           77           78 
##  1.144509191  1.161959330  1.179409469  1.196859608  1.214309747  1.231759887 
##           79           80           81           82           83           84 
##  1.249210026  1.266660165  1.284110304  1.301560443  1.319010582  1.336460722 
##           85           86           87           88           89           90 
##  1.353910861  1.371361000  1.388811139  1.406261278  1.423711417  1.441161557 
##           91           92           93           94           95           96 
##  1.458611696  1.476061835  1.493511974  1.510962113  1.528412253  1.545862392 
##           97           98           99          100 
##  1.563312531  1.580762670  1.598212809  1.615662948
```

```
##
## $se.fit
##          1          2          3          4          5          6          7
## 0.29268408 0.28809204 0.28350881 0.27893482 0.27437054 0.26981645 0.26527309
##          8          9         10         11         12         13         14
## 0.26074101 0.25622081 0.25171314 0.24721867 0.24273814 0.23827234 0.23382211
##         15         16         17         18         19         20         21
## 0.22938836 0.22497206 0.22057426 0.21619609 0.21183876 0.20750360 0.20319201
##         22         23         24         25         26         27         28
## 0.19890553 0.19464582 0.19041468 0.18621405 0.18204605 0.17791297 0.17381729
##         29         30         31         32         33         34         35
## 0.16976174 0.16574924 0.16178301 0.15786654 0.15400361 0.15019837 0.14645530
##         36         37         38         39         40         41         42
## 0.14277930 0.13917568 0.13565021 0.13220915 0.12885924 0.12560780 0.12246266
##         43         44         45         46         47         48         49
## 0.11943222 0.11652542 0.11375175 0.11112118 0.10864411 0.10633125 0.10419355
##         50         51         52         53         54         55         56
## 0.10224199 0.10048742 0.09894033 0.09761056 0.09650712 0.09563783 0.09500912
##         57         58         59         60         61         62         63
## 0.09462579 0.09449082 0.09460528 0.09496826 0.09557694 0.09642665 0.09751111
##         64         65         66         67         68         69         70
## 0.09882257 0.10035215 0.10209004 0.10402579 0.10614859 0.10844745 0.11091142
##         71         72         73         74         75         76         77
## 0.11352976 0.11629204 0.11918824 0.12220885 0.12534488 0.12858788 0.13192997
##         78         79         80         81         82         83         84
## 0.13536379 0.13888256 0.14247997 0.14615022 0.14988796 0.15368827 0.15754661
##         85         86         87         88         89         90         91
## 0.16145882 0.16542110 0.16942991 0.17348204 0.17757451 0.18170462 0.18586984
##         92         93         94         95         96         97         98
## 0.19006786 0.19429657 0.19855400 0.20283834 0.20714792 0.21148120 0.21583675
##         99        100
## 0.22021325 0.22460948
##
## $residual.scale
## [1] 1
```

```
#confidence intervals
low <- exp(prd$fit - qnorm(.975) * prd$se.fit)
high <- exp (prd$fit + qnorm(.975) * prd$se.fit)

# plot
plot(y = exp(prd$fit), x = x1_new$x1, xlab = 'x1',
ylab = 'Expected count', cex.axis = 1.5, cex.lab = 1.5,
ylim = c(min(low), max(high)), type = 'l')
lines(x = x1_new$x1, y = low, lty = 2)
lines(x = x1_new$x1, y = high, lty = 2)
```



Predict the expected count, ± 95% confidence intervals, of levels "a" and "b", assuming x1 is fixed at it's mean.

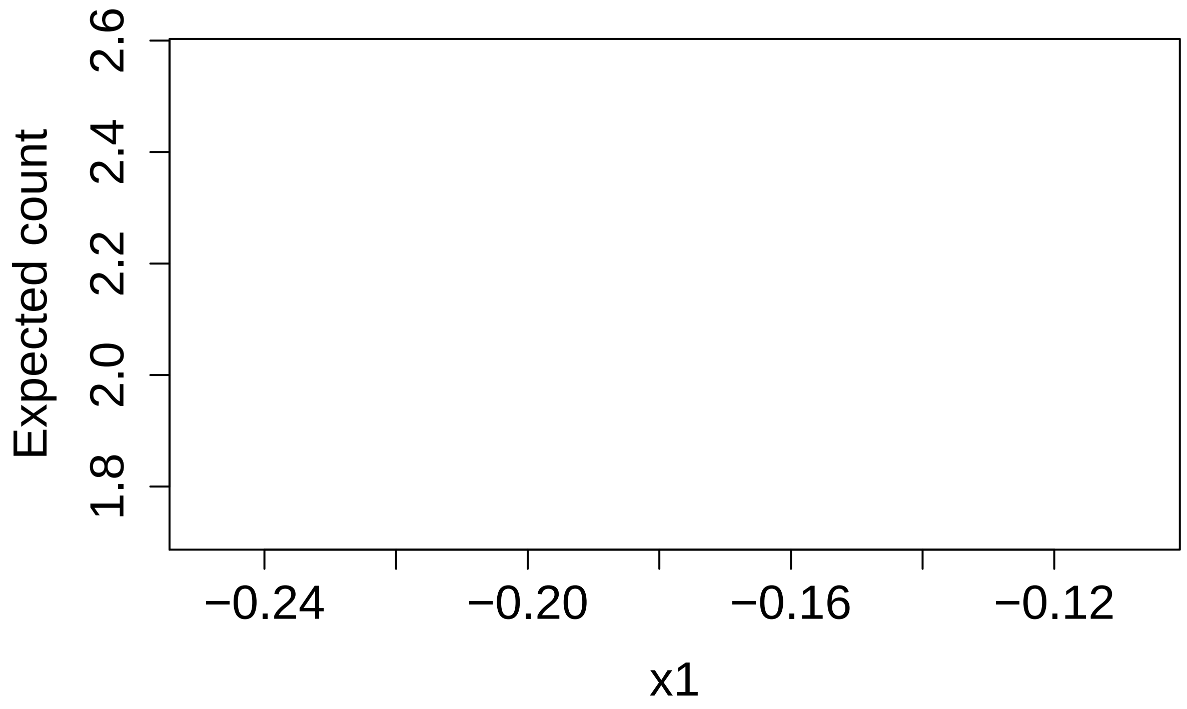
```
x1_new2 <- data.frame(
  x1 = mean(hw4$x1),
  x2 = factor('a', levels = c('a','b')))
# x1 is mean and x2 is b
x1_new2b <- data.frame(
  x1 = mean(hw4$x1),
  x2 = factor('b', levels = c('a','b')))



# predicted expected count
prd2a <- predict.glm(object= fit, newdata = x1_new2, type = 'link', se.fit = T)
prd2a
```

```
## $fit
##        1
## 1.874758
##
## $se.fit
## [1] 0.05530042
##
## $residual.scale
## [1] 1
```

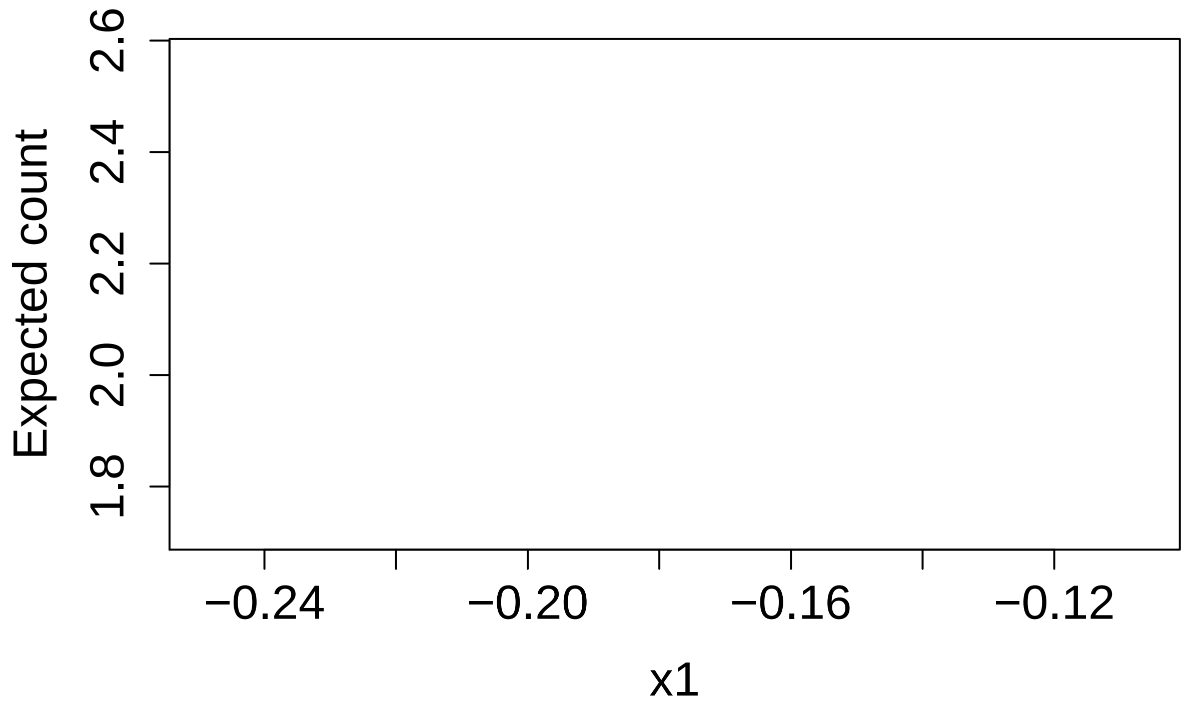
```r
prd2b <- predict.glm(object= fit, newdata = x1_new2b, type = 'link', se.fit = T)
prd2b
```

```
## $fit
##         1
## 0.7431383
##
## $se.fit
## [1] 0.1022439
##
## $residual.scale
## [1] 1
```

```r
#confidence intervals
low <- exp(prd2a$fit - qnorm(.975) * prd2a$se.fit)
high <- exp (prd2a$fit + qnorm(.975) * prd2a$se.fit)

low <- exp(prd2b$fit - qnorm(.975) * prd2b$se.fit)
high <- exp (prd2b$fit + qnorm(.975) * prd2b$se.fit)

# plot
plot(y = exp(prd2a$fit), x = x1_new2$x1, xlab = 'x1',
ylab = 'Expected count', cex.axis = 1.5, cex.lab = 1.5,
ylim = c(min(low), max(high)), type = 'l')
lines(x = x1_new2$x1, y = low, lty = 2)
lines(x = x1_new2$x1, y = high, lty = 2)
```



```
plot(y = exp(prd2b$fit), x = x1_new2b$x1, xlab = 'x1',
ylab = 'Expected count', cex.axis = 1.5, cex.lab = 1.5,
ylim = c(min(low), max(high)), type = 'l')
lines(x = x1_new2b$x1, y = low, lty = 2)
lines(x = x1_new2b$x1, y = high, lty = 2)
```


Expected count
1.8
2.0
2.2
2.4
2.6
−0.24
−0.20
−0.16
−0.12
x1