Kingsbury_Homework_4

Joe Kingsbury

3/17/2021

Question 1

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

```r
dat <- read.csv("Homework 4 Data.csv")
head(dat)
```

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

```
##
## Call:
## glm(formula = y ~ x1 * x2, family = poisson, data = dat)
##
## 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
```

Question 2

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

```r
betas <- coef(fit)
y_1 <- betas[1] + betas[2] * 1 + betas[3] + betas[4] * 1
y_2 <- betas[1] + betas[2] * 2 + betas[3] + betas[4] * 2
exp(y_2) / exp(y_1)
```

```
## (Intercept)
##    1.460856
```

```r
exp(betas[2] + betas[4])
```

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

```r
(exp(betas[2] + betas[4]) - 1) * 100
```

```
##       x1
## 46.08559
```

The expected count of Y increased by %46.09 for each single unit increase in x1 when taking into account x2 = b.

Question 3

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

```r
y_a <- betas[1] + betas[2] * 1
y_b <- betas[1] + betas[2] * 1 + betas[3] + betas[4] * 1
log(exp(y_b) / exp(y_a))
```

```
## (Intercept)
##  -0.5682285
```

```r
betas[3] + betas[4]
```

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

```
(exp(betas[3] + betas[4]) -1) * 100
```

```
##       x2b
## -43.34718
```

When x1 is held at 1 there is a %43.35 difference between the expected count of y when comparing category a and category b.

Question 4

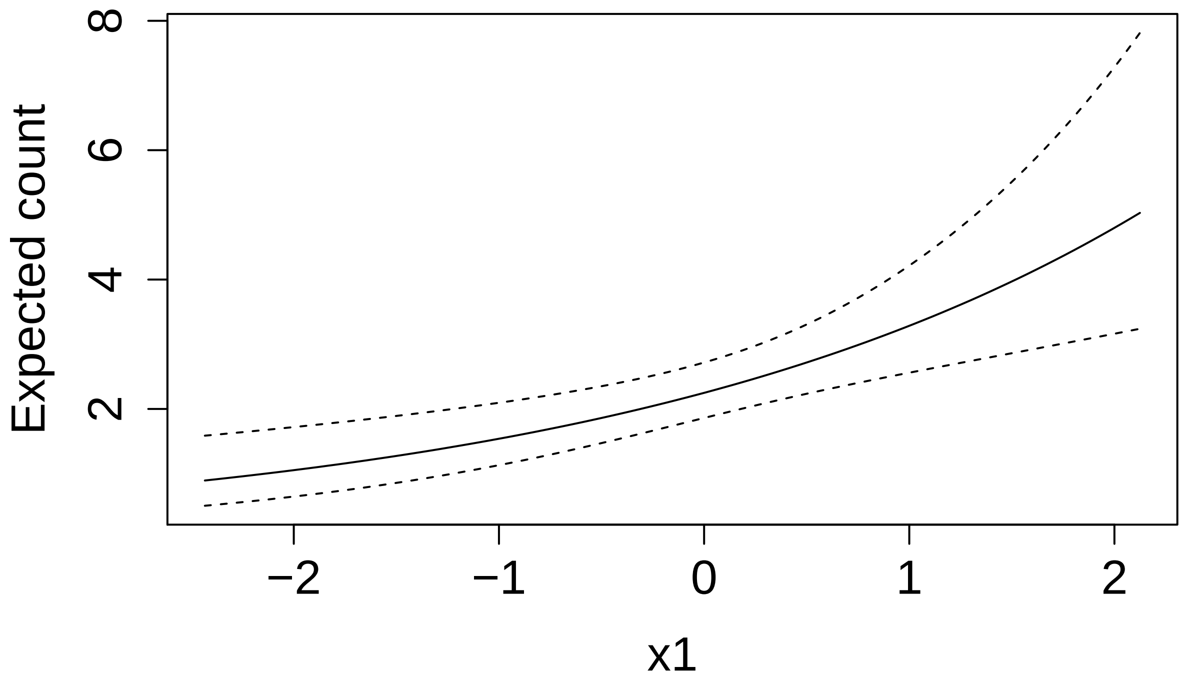
Predict the expected count, 95% confidence intervals, over the observed range of values of x1, assuming x2 is fixed at level "b".

```
#Create a new dataframe of predicted values
data2 <- data.frame(
x1 = seq(min(dat$x1), max(dat$x1), length.out = 100),
x2 = factor(x = rep('b', times = 100),
levels = c('a','b')))
head(data2)
```

```
##          x1 x2
## 1 -2.433575  b
## 2 -2.387535  b
## 3 -2.341495  b
## 4 -2.295455  b
## 5 -2.249415  b
## 6 -2.203376  b
```

```
prd1 <- predict.glm(object = fit, newdata = data2, type = 'link', se.fit = T)
low <- exp(prd1$fit - qnorm(0.975) * prd1$se.fit)
high <- exp(prd1$fit + qnorm(0.975) * prd1$se.fit)

plot(y = exp(prd1$fit), x = data2$x1, xlab = 'x1',
  ylab = 'Expected count', cex.axis = 1.5, cex.lab = 1.5,
  ylim = c(min(low), max(high)), type = 'l')
  lines(x = data2$x1, y = low, lty = 2)
  lines(x = data2$x1, y = high, lty = 2)
```



Question 5

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

```
#Predicted count of level "a" with x1 fixed at it's mean
y_a_prd <- betas[1] + betas[2] * mean(dat$x1)
exp(y_a_prd)
```

```
## (Intercept)
##    6.519241
```

```
#Predicted count of level "b" with x1 fixed at it's mean
y_b_prd <- betas[1] + betas[2] * mean(dat$x1) + betas[3] + betas[4] * mean(dat$x1)
exp(y_b_prd)
```

```
## (Intercept)
##    2.102524
```

```
#Create a new datafram with x1 at it's mean and level a and b present
data3 <- data.frame(
  x1 = rep(mean(dat$x1), times = 2),
  x2 = c('a', 'b'))

y_prd <- predict.glm(object = fit, newdata = data3, type = 'link', se.fit = T )
```

```r
y_low <- exp(y_prd$fit - qnorm(0.975) * y_prd$se.fit)
y_high <- exp(y_prd$fit + qnorm(0.975) * y_prd$se.fit)
```

Confidence Interval for "a"

```r
Name <- c("Lower Limit", "Upper Limit")
aConfidenceInterval <- cbind.data.frame(y_low[1:1], y_high[1:1])
names(aConfidenceInterval) <- Name
aConfidenceInterval
```

```
##   Lower Limit Upper Limit
## 1    5.849587    7.265556
```

Confidence Interval for "b"

```r
bConfidenceInterval <- cbind.data.frame(y_low[2:2], y_high[2:2])
names(bConfidenceInterval) <- Name
bConfidenceInterval
```

```
##   Lower Limit Upper Limit
## 2    1.720722    2.569041
```