Logistic regression for binary data

[brouwern@gmail.com](mailto:brouwern@gmail.com)

December 2, 2016

Table of Contents

# Alternative kinds of regression

## Regression with "binary data""

### "Binary"=

* **0/1**
* **either/or**
* **yes/no**
* **alive/dead**
* **flowered / didnt flower**
* **eaten by deer / not eaten by**
* **caught something in trap / didn't catch something**
* **shot a deer/didn't shoot a deer**

## Two forms of binary data

### Binary data Form 1:

#### (# of "events")/(# of "trials")

* (# of dead tadpoles) / (# exposed to pesticide)
* (# of eaten plants) / (# of plants in forest)
* (# of deer shot) / (# of hunters in forest)

Several people in class have this kind of data.

#### Requires that the denominator is known

#### Often used when you have a *categorical* predictor

#### Often used for experiments that generate binary data

### Binary data Form 2:

#### Did the event happen? ~ predictor variable

* Did you shoot a deer? ~ distance from forest edge
* Did the Trillium flower? ~ size of the plant
* Was the Trillium eaten? ~ number of deer in forest
* Did the person get the flu? ~ concentration of vaccine given

A few people in class might have this kind of data. Shows up often in field studies.

## Binary data Example 1: (# of "events")/(# of "trials")

### Experiment: effect of cold exposure on fish

Whitlock & Shulter logistic regression example page 568

## Load data from page 568  
  
### Original data has a sample size of 40  
df1.n40 <- data.frame(died.YN = c(11,24,29,38),  
 lived.YN = c(29,16,11,2),  
 exposure.min = c(3,8,12,18))  
  
df1.n40$total.fish <- df1.n40$died.YN +df1.n40$lived.YN  
  
#reduce same size to 20  
#make change to data for illustration purposes  
df1.n20 <- df1.n40  
df1.n20$died.YN <- round(df1.n20$died.YN/2,0)  
df1.n20$lived.YN <- round(df1.n20$lived.YN/2,0)  
df1.n20$total.fish <- df1.n20$died.YN +df1.n20$lived.YN

### Look at dataframe

df1.n20

## died.YN lived.YN exposure.min total.fish  
## 1 6 14 3 20  
## 2 12 8 8 20  
## 3 14 6 12 20  
## 4 19 1 18 20

## Common way to think about binary data: percentages

* Binary data often converted to percentage
* Divide # of events by # of trials
* This can cause problems

Calculate perentage mortality for fish data

percent.died <- df1.n20$died.YN/df1.n20$total.fish  
  
df1.n20 <- cbind(percent.died, df1.n20)

Look at new percentage data

df1.n20

## percent.died died.YN lived.YN exposure.min total.fish  
## 1 0.30 6 14 3 20  
## 2 0.60 12 8 8 20  
## 3 0.70 14 6 12 20  
## 4 0.95 19 1 18 20

## The problem with percentages from binary data

### Problem 1: they hid the sample size

* What if I said "50% of people who stand in front of a microrwave while warming up their coffee get cancer, and I have data to prove it."
* What would be a key feature of this study? ... Sample size!
* What if my data had n = 2? (1 got cancer)
* What if my data had n = 100? (50 got cancer)
* Both data have caner rate = 50%
* Which dataset is more reliable?

### Problem 2: people who use them often incorrectly calcualte confidence intervals (CIs)

#### Model percent.died vs. exposure to cold

Use standard regression w/ lm()

lm.percent <- lm(percent.died ~ exposure.min,   
 data = df1.n20)

#### Look at model output

Use summary(...)

summary(lm.percent)

##   
## Call:  
## lm(formula = percent.died ~ exposure.min, data = df1.n20)  
##   
## Residuals:  
## 1 2 3 4   
## -0.03354 0.05683 -0.01087 -0.01242   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.207764 0.050907 4.081 0.0551 .  
## exposure.min 0.041925 0.004377 9.578 0.0107 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0481 on 2 degrees of freedom  
## Multiple R-squared: 0.9787, Adjusted R-squared: 0.968   
## F-statistic: 91.74 on 1 and 2 DF, p-value: 0.01073

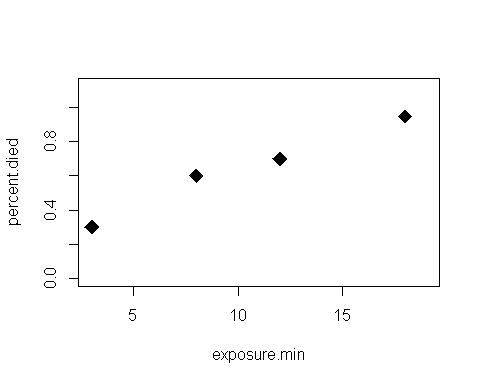
* As exposure time increase, the percentage of fish that die increases.
* What number tells you this?

#### Plot the data

newdat <- data.frame(exposure.min = 3:20)  
ci.out <- predict(lm.percent,  
 interval = "confidence",  
 newdata = newdat)

#### Plot the data with confidence intervals

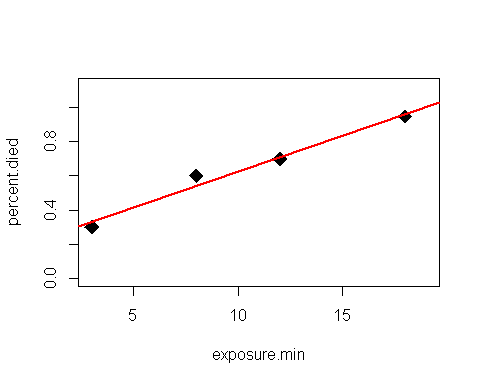
par(mfrow = c(1,1))  
plot(percent.died ~ exposure.min, data = df1.n20,   
 pch = 18, cex = 2,  
 xlim = c(3, 19),  
 ylim = c(0,1.125))



Ok, as exposure to cold increases, mortality increase

#### Plot the percentage data w/ the regression line

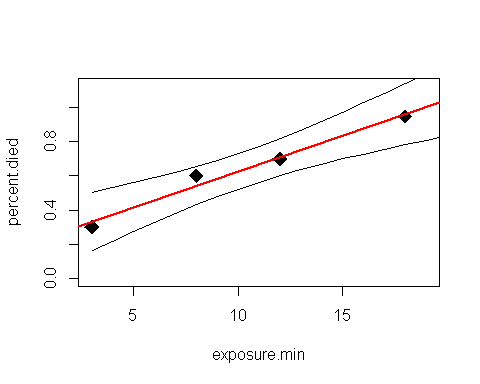
par(mfrow = c(1,1))  
plot(percent.died ~ exposure.min, data = df1.n20,   
 pch = 18, cex = 2,  
 xlim = c(3, 19),  
 ylim = c(0,1.125))  
abline(lm.percent, col = 2, lwd = 2)



Nice regression line showing trend

#### Plot the data w/regression line & 95% CI

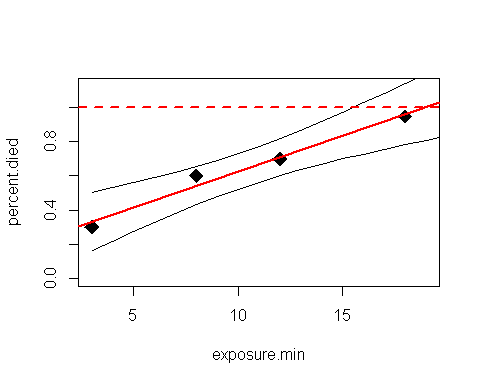
par(mfrow = c(1,1))  
plot(percent.died ~ exposure.min, data = df1.n20,   
 pch = 18, cex = 2,  
 xlim = c(3, 19),  
 ylim = c(0,1.125))  
abline(lm.percent, col = 2, lwd = 2)  
points(ci.out[,"lwr"] ~ newdat$exposure.min,   
 type = "l")  
points(ci.out[,"upr"] ~ newdat$exposure.min, type = "l")



Fancy confidnece interval around trend. Nice, but...

#### Plot the data w/regression line, 95% CI, and refernece line for percent.died = 1

plot(percent.died ~ exposure.min, data = df1.n20,   
 pch = 18, cex = 2,  
 xlim = c(3, 19),  
 ylim = c(0,1.125))  
abline(lm.percent, col = 2, lwd = 2)  
points(ci.out[,"lwr"] ~ newdat$exposure.min,   
 type = "l")  
points(ci.out[,"upr"] ~ newdat$exposure.min, type = "l")  
abline(h = 1, col = 2, lwd = 2, lty = 2)



* What does it mean the the confidence interval crosses 1.0?
* Subtle point: if your model can make confidence intervals for data points that don't make sense, something is wrong with the model!

## Correct analysis of binary data

### A correct analysis uses a model that

#### 1)Includes information on the sample size

#### 2)Properly models the binary nature of the data

#### This requires "logistic regression"

#### This is a type of "generalized linear model"

## Coding a logistic regression GLM

* **uses R's glm() function**
* **requires a statement of the "family" of the model**
* **here family = "binomial"**
* **Also need to indicate sample size (n)**
* **use "weights = " argument**
* **here, weights = total.fish**

## Code to run the logistic regression model model

glm1 <- glm(percent.died ~ exposure.min,   
 data = df1.n20,  
   
 #model family:  
 family = "binomial",  
   
 #sample size  
 weights = total.fish)

#### Logistic regression GLM output

Use summary() command just as for regular regression

summary(glm1)

##   
## Call:  
## glm(formula = percent.died ~ exposure.min, family = "binomial",   
## data = df1.n20, weights = total.fish)  
##   
## Deviance Residuals:   
## 1 2 3 4   
## 0.01735 0.30995 -0.62850 0.45772   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.5280 0.5613 -2.722 0.006479 \*\*   
## exposure.min 0.2241 0.0579 3.870 0.000109 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 21.04467 on 3 degrees of freedom  
## Residual deviance: 0.70089 on 2 degrees of freedom  
## AIC: 16.691  
##   
## Number of Fisher Scoring iterations: 4

* This looks almost the sames as for regular regression
* BUT - the "slope" parameter for a GLM only tells you the overall direction of a trend (positive or negative)
* The slope is on a special "logistic regression" scale

#### Plot output of model

* Logistic regression output is on a special scale
* Why this is is beyond this class: it has to do with logs :(
* R's "predict" function converts from this "special" to a percentage/probability scale
* This special scale is called the "link" scale
* The normal scale we can interpret is the "response" scale
* That is, the scale that the original response variable was measured on

Get predictions from our GLM

#Predictions from GLM  
newdat <- data.frame(exposure.min = 3:20)  
se.outlink <- predict(glm1,  
 se.fit = T,  
 type = "link",  
 newdata = newdat)  
  
library(arm)

## Loading required package: MASS

## Loading required package: Matrix

## Loading required package: lme4

##   
## Attaching package: 'lme4'

## The following object is masked from 'package:stats':  
##   
## sigma

##   
## arm (Version 1.8-6, built: 2015-7-7)

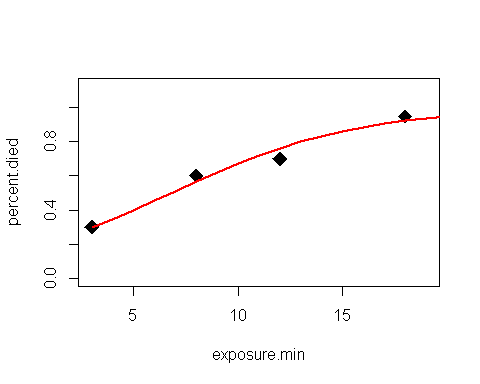
## Working directory is C:/Users/lisanjie2/Desktop/TEACHING/1\_STATS\_CalU/1\_STAT\_CalU\_2016\_by\_NLB/Lecture/Unit3\_regression/week\_after\_thanksgiving

glm.cis <- data.frame(predictions = invlogit(se.outlink$fit),  
 lwr = invlogit(se.outlink$fit - 1.96\*se.outlink$se.fit),  
 upr = invlogit(se.outlink$fit + 1.96\*se.outlink$se.fit))  
  
glm.cis <- cbind(glm.cis,newdat)

#### Plot the output of the GLM:

##### Data w/regression line

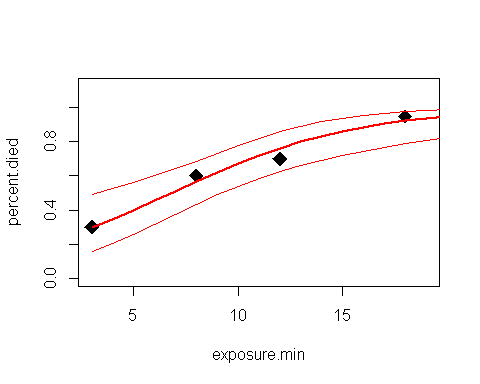
plot(percent.died ~ exposure.min, data = df1.n20,   
 pch = 18, cex = 2,  
 xlim = c(3, 19),  
 ylim = c(0,1.125))  
points(predictions ~ exposure.min,   
 type = "l", data = glm.cis, col = 2, lwd = 2)



#### Plot the output of the GLM:

##### Data, line, confidence intervals

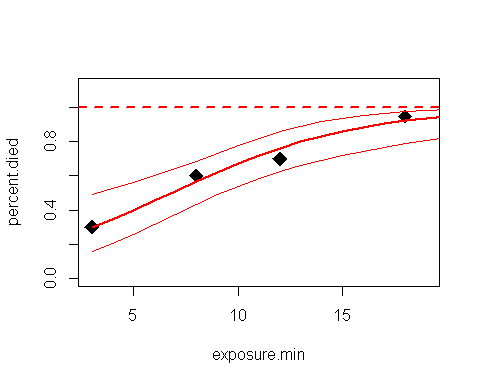
plot(percent.died ~ exposure.min, data = df1.n20,   
 pch = 18, cex = 2,  
 xlim = c(3, 19),  
 ylim = c(0,1.125))  
points(predictions ~ exposure.min,   
 type = "l", data = glm.cis, col = 2, lwd = 2)  
points(upr ~ exposure.min,   
 type = "l", data = glm.cis, col = 2)  
points(lwr ~ exposure.min,   
 type = "l", data = glm.cis, col = 2)



#### Plot the output of the GLM:

##### Data, line, CIs, and reference line at 1.0

plot(percent.died ~ exposure.min, data = df1.n20,   
 pch = 18, cex = 2,  
 xlim = c(3, 19),  
 ylim = c(0,1.125))  
points(predictions ~ exposure.min,   
 type = "l", data = glm.cis, col = 2, lwd = 2)  
points(upr ~ exposure.min,   
 type = "l", data = glm.cis, col = 2)  
points(lwr ~ exposure.min,   
 type = "l", data = glm.cis, col = 2)  
  
abline(h = 1, col = 2, lwd = 2, lty = 2)



## Binary data Example 1: Did event happen? ~ predictor