P8131_hw2

Apoorva Srinivasan

2/19/2019

```
library(broom)
library(tidyverse)
```

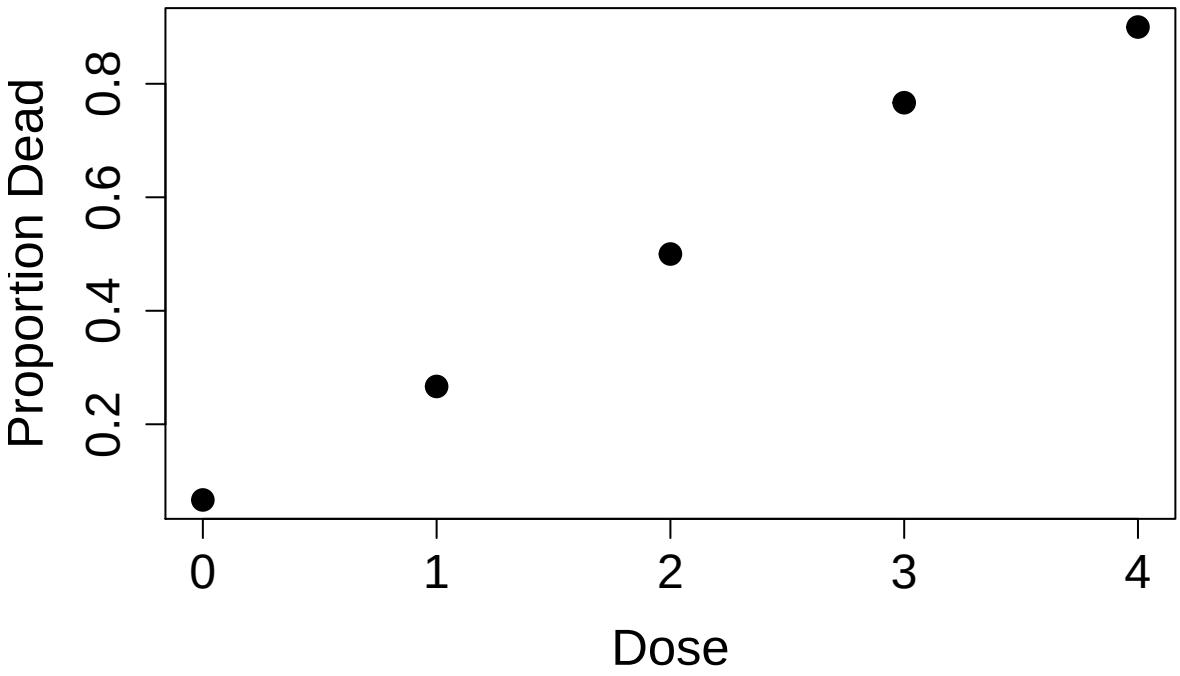
```
## -- Attaching packages ----------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.8
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts -------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Problem 1

```
dose = c(0,1,2,3,4)
number = c(30,30,30,30,30)
dead = c(2,8,15,23,27)
data = data.frame(dose, number, dead)
plot(data$dose,data$dead/data$number,xlab='Dose',ylab='Proportion Dead',cex=1.5,pch=19,cex.lab=1.6,cex.a
```




```
x=data$dose
y=data$dead
m = data$number
response = cbind(y,m-y)
```

```r
glm_logit=glm(response~x, family=binomial(link='logit'))
summary(glm_logit) # wald test of coefficients
```

```
##
## Call:
## glm(formula = response ~ x, family = binomial(link = "logit"))
##
## Deviance Residuals:
##       1        2        3        4        5
## -0.4510   0.3597   0.0000   0.0643  -0.2045
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.3238     0.4179  -5.561 2.69e-08 ***
## x             1.1619     0.1814   6.405 1.51e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 64.76327  on 4  degrees of freedom
## Residual deviance:  0.37875  on 3  degrees of freedom
## AIC: 20.854
##
## Number of Fisher Scoring iterations: 4
```

```r
glm_probit=glm(response~x, family=binomial(link='probit'))
summary(glm_probit)
```

```
##
## Call:
## glm(formula = response ~ x, family = binomial(link = "probit"))
##
## Deviance Residuals:
##        1         2         3         4         5
## -0.35863   0.27493   0.01893   0.18230  -0.27545
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.37709    0.22781  -6.045 1.49e-09 ***
## x            0.68638    0.09677   7.093 1.31e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 64.76327  on 4  degrees of freedom
## Residual deviance:  0.31367  on 3  degrees of freedom
## AIC: 20.789
##
## Number of Fisher Scoring iterations: 4
```

```r
glm_cloglog = glm(response~x, family=binomial(link='cloglog'))
summary(glm_cloglog)
```

```
##
## Call:
## glm(formula = response ~ x, family = binomial(link = "cloglog"))
##
## Deviance Residuals:
##       1        2        3        4        5
## -1.0831   0.2132   0.4985   0.5588  -0.6716
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.9942     0.3126  -6.378 1.79e-10 ***
## x             0.7468     0.1094   6.824 8.86e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 64.7633  on 4  degrees of freedom
## Residual deviance:  2.2305  on 3  degrees of freedom
## AIC: 22.706
##
## Number of Fisher Scoring iterations: 5
```

```
deviance =sum(residuals(glm_logit,type='deviance')^2)
vcov(glm_logit)
```

```
##             (Intercept)           x
## (Intercept)  0.17463024 -0.06582336
## x           -0.06582336  0.03291168
```

```
beta =glm_logit$coefficients[2]
se=sqrt(vcov(glm_logit)[2,2])
CI = beta+c(qnorm(0.025),-qnorm(0.025))*se
pi_hat = predict(glm_logit, data.frame(x=0.01), se.fit=TRUE, type = 'response')
p_hat = pi_hat[[1]]
```

```
glm_table <- function(fit){

  beta = fit %>%
    broom::tidy() %>%
    filter(term == 'x') %>%
    pull(estimate)


  deviance = deviance(fit)

  se = fit %>%
    broom::tidy() %>%
    pull(std.error) %>% .[2]
  CI_high = beta - se * qnorm(0.025)
CI_low = beta + se * qnorm(0.025)



  pi_hat = predict(fit, data.frame(x = 0.01), se.fit = TRUE, type = 'response')
```

```
p_hat = pi_hat[[1]]

  tibble(beta, CI_low, CI_high, deviance, p_hat)
}

bind_rows(
  glm_table(glm_logit),
  glm_table(glm_probit),
  glm_table(glm_cloglog)
  ) %>%
mutate(model = c('logit', 'probit', 'c-loglog')) %>%
select(model, everything()) %>%
knitr::kable(digits = 4)
```

model	beta	CI_low	CI_high	deviance	p_hat
logit	1.1619	0.8063	1.5175	0.3787	0.0901
probit	0.6864	0.4967	0.8760	0.3137	0.0853
c-loglog	0.7468	0.5323	0.9613	2.2305	0.1282

Comments: We observe three different estimates for the 3 models with different link functions. Probit and logit have lower comparable deviances while cloglog has a much higher deviance indicating a poorer fit.

ii) Estimate LD50 with 90% CI based on the THREE models.

```
ld50_ci = function(fit, alpha=0.1){
  beta0 = fit$coefficients[1]
  beta1 = fit$coefficients[2]
  betacov = vcov(fit)
  x0fit = -beta0/beta1
  varx0 = betacov[1,1]/(beta1^2) + betacov[2,2]*(beta0^2)/(beta1^4) - 2*betacov[1,2]*beta0/(beta1^3)
  tibble(estiamte = exp(x0fit),
      CI_low = exp(x0fit + qnorm(alpha/2) * sqrt(varx0)),
      CI_high = exp(x0fit - qnorm(alpha/2) * sqrt(varx0))
     )
}


probit =
  ld50_ci(glm_probit) %>% mutate(model = 'probit') %>%
  select(model, everything())
probit
```

```
## # A tibble: 1 x 4
##   model  estiamte CI_low CI_high
##   <chr>     <dbl>  <dbl>   <dbl>
## 1 probit     7.44   5.58    9.90
```

```
logit =
  ld50_ci(glm_logit) %>% mutate(model = 'logit') %>%
  select(model, everything())
logit
```

```
## # A tibble: 1 x 4
##   model estiamte CI_low CI_high
##   <chr>    <dbl>  <dbl>   <dbl>
## 1 logit     7.39   5.51    9.91
```

```
# c-log-log model
log(-log(0.5))
```

```
## [1] -0.3665129
```

```
alpha = 0.1
fit = glm_cloglog
beta0 = fit$coefficients[1]
beta1 = fit$coefficients[2]
betacov = vcov(fit)
x0fit = (-beta0 + log(-log(0.5)) ) / beta1
varx0 = betacov[1,1]/(beta1^2) + betacov[2,2]*((-log(-log(0.5))+beta0)^2)/(beta1^4) - 2*betacov[1,2]*(-

cloglog =
  tibble(estiamte = exp(x0fit),
      CI_low = exp(x0fit + qnorm(alpha/2) * sqrt(varx0)),
      CI_high = exp(x0fit - qnorm(alpha/2) * sqrt(varx0))
      ) %>%
  mutate(model = 'cloglog') %>% select(model, everything())
cloglog
```

```
## # A tibble: 1 x 4
##   model   estiamte CI_low CI_high
##   <chr>      <dbl>  <dbl>   <dbl>
## 1 cloglog     8.84   6.53    12.0
```

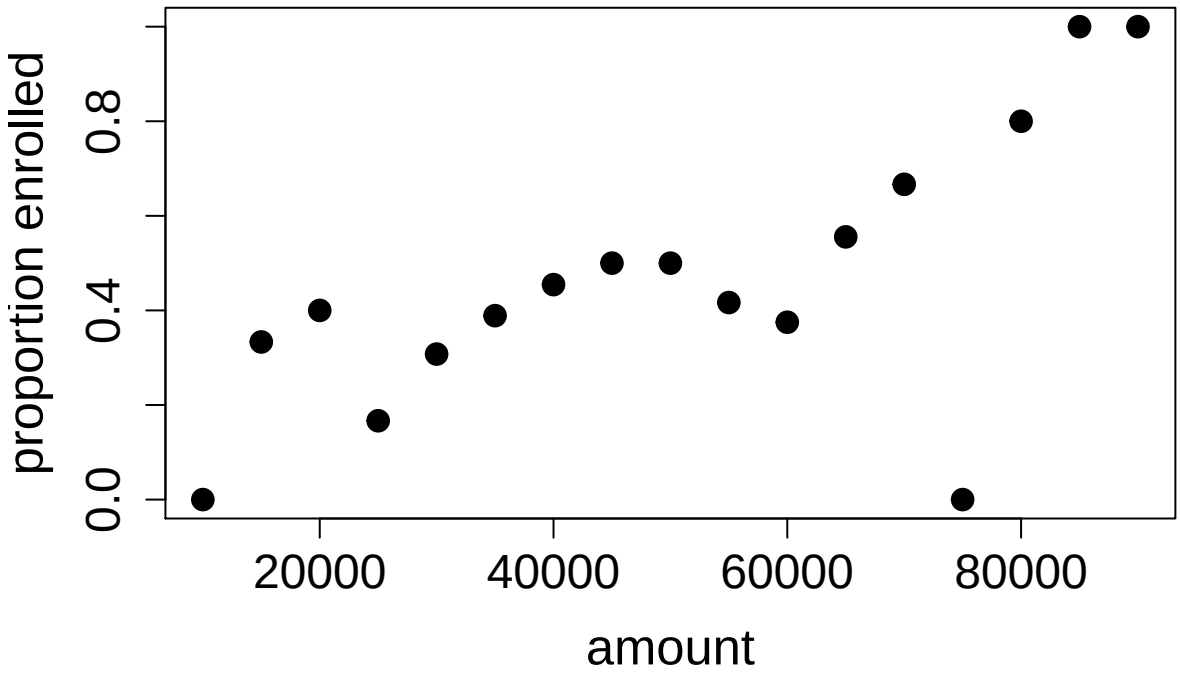
```
rbind(logit, probit, cloglog) %>%
  knitr::kable()
```

model	estiamte	CI_low	CI_high
logit	7.389056	5.509632	9.909583
probit	7.435830	5.582588	9.904289
cloglog	8.841249	6.526261	11.977407

Problem 2

```
amount = c(10000, 15000, 20000, 25000, 30000, 35000, 40000, 45000, 50000, 55000, 60000, 65000,70000, 75
offers = c(4,6,10,12, 39, 36, 22, 14, 10, 12, 8,9, 3, 1, 5, 2, 1)
enrolls = c(0,2,4,2, 12, 14, 10, 7, 5, 5, 3, 5, 2, 0,4,2,1)
mph_data = data.frame(amount, offers, enrolls)

plot( mph_data$amount,mph_data$enrolls/mph_data$offers, xlab='amount',ylab='proportion enrolled',cex=1.5
```




```
fit_glm = glm(cbind(enrolls, offers - enrolls)~amount, family = binomial(link = 'logit'))
summary(fit_glm)
```

```
##
## Call:
## glm(formula = cbind(enrolls, offers - enrolls) ~ amount, family = binomial(link = "logit"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4735  -0.6731   0.1583   0.5285   1.1275
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.648e+00  4.214e-01  -3.910 9.25e-05 ***
## amount       3.095e-05  9.680e-06   3.197  0.00139 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 21.617  on 16  degrees of freedom
## Residual deviance: 10.613  on 15  degrees of freedom
## AIC: 51.078
##
## Number of Fisher Scoring iterations: 4
```

```
dev = deviance(fit_glm)
pval = 1 - pchisq(dev, 15);  pval
```

```
## [1] 0.7795345
```

i) Since p value is much greater than 0.05, we conclude that it fits the data well

```r
beta1 = fit_glm %>% broom::tidy() %>% filter(term == 'amount') %>% pull(estimate)
beta0 = fit_glm %>% broom::tidy() %>% filter(term == '(Intercept)') %>% pull(estimate)

std_error = fit_glm %>% broom::tidy() %>% filter(term == 'amount') %>% pull(std.error)
std_error1 = fit_glm %>% vcov %>% .[2,2] %>% sqrt
std_error == std_error1
```

```
## [1] TRUE
```

```r
# 95% CI for beta_1
beta1_result =
  tibble(term = 'beta1',
         estimate = beta1,
         CI_low = beta1 - std_error1 * qnorm(1-0.05/2),
         CI_high = beta1 + std_error1 * qnorm(1-0.05/2)
        )

# 95% CI for beta_0
std_error0 = fit_glm %>% broom::tidy() %>% filter(term == '(Intercept)') %>% pull(std.error)
beta0_result =
  tibble(term = 'beta0',
         estimate = beta0,
         CI_low = beta0 - std_error0 * qnorm(1-0.05/2),
         CI_high = beta0 + std_error0 * qnorm(1-0.05/2)
        )

rbind(beta0_result, beta1_result) %>% knitr::kable()
```

term	estimate	CI_low	CI_high
beta0	-1.647638	-2.473645	-0.8216318
beta1	0.000031	0.000012	0.0000499

ii) The estimated beta_1 is 0.031, and the predictor (Amount) was fitted on the scale of thousand dollars. Therefore, the log odds of enrollment would increase by 0.031 with 1,000 dollars' increase in the scholarship.

```r
pi = 0.4
x0 = (log(pi/(1 - pi)) - beta0) / beta1
beta0 = fit_glm %>% broom::tidy() %>% filter(term == '(Intercept)') %>% pull(estimate)
beta1 = fit_glm %>% broom::tidy() %>% filter(term == 'amount') %>% pull(estimate)
betacov = vcov(fit_glm)
varx0 = betacov[1,1]/(beta1^2) + betacov[2,2]*((-log(4/6)+beta0)^2)/(beta1^4) - 2*betacov[1,2]*(-log(4/6
alpha = 0.05
result =
  tibble(estimate = x0,
         CIL = x0 + qnorm(alpha/2) * sqrt(varx0),
         CIR = x0 - qnorm(alpha/2) * sqrt(varx0)
         )

result %>% knitr::kable()
```

estimate	CIL	CIR
40134.29	30583.04	49685.53

iii) From the results above, we can see that we should provide \$40,134 for 40% yield rate