Lec13_1

Jiansong Xu

February 27, 2019

1

a.

```
shuttle <- read.csv("challenger.csv")
probit.fit <- glm(O.ring/Number ~ Temp, family = binomial(link = "probit"), weights = Number, data = shuttle
)
summary(probit.fit)
```

```
##
## Call:
## glm(formula = O.ring/Number ~ Temp, family = binomial(link = "probit"),
##     data = shuttle, weights = Number)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.97786  -0.80338  -0.53608  -0.01783   2.65281
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.23452    1.58759   1.407   0.1593
## Temp        -0.05550    0.02381  -2.331   0.0197 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24.230  on 22  degrees of freedom
## Residual deviance: 18.311  on 21  degrees of freedom
## AIC: 35.872
##
## Number of Fisher Scoring iterations: 5
```

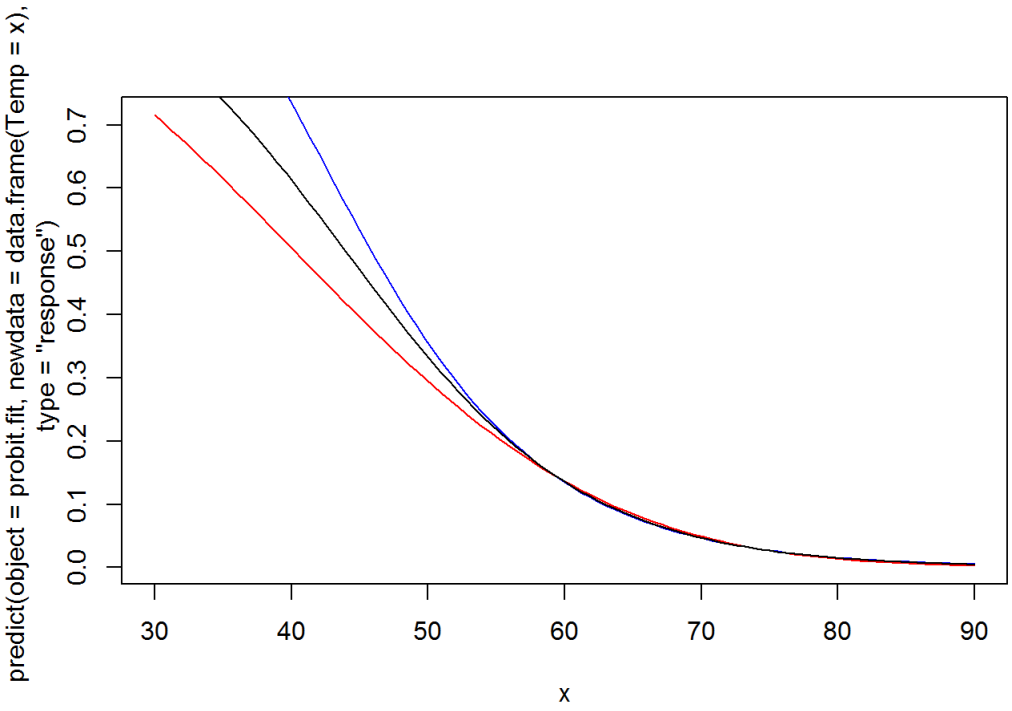
b.

```
log.fit <- glm(O.ring/Number ~ Temp, family = binomial(link = "cloglog"), weights = Number, data = shuttle)
summary(log.fit)
```

```
##
## Call:
## glm(formula = O.ring/Number ~ Temp, family = binomial(link = "cloglog"),
##     data = shuttle, weights = Number)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.94590  -0.77952  -0.54366  -0.04815   2.64743
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.71422    2.78092   1.695   0.0900 .
## Temp        -0.11076    0.04332  -2.557   0.0106 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24.230  on 22  degrees of freedom
## Residual deviance: 18.014  on 21  degrees of freedom
## AIC: 35.574
##
## Number of Fisher Scoring iterations: 5
```

c.

```
fit <- glm(O.ring/Number ~ Temp, family = binomial(link = "logit"), weights = Number, data = shuttle)
curve(expr = predict(object = probit.fit, newdata = data.frame(Temp = x), type = "response"), col = "red", x
lim = c(30, 90))
curve(expr = predict(object = log.fit, newdata = data.frame(Temp = x), type = "response"), col = "blue", xli
m = c(30, 90), add = TRUE)
curve(expr = predict(object = fit, newdata = data.frame(Temp = x), type = "response"), col = "black", xlim =
c(30, 90), add = TRUE)
```



d.

```
alpha = 0.05
##Probit
probit.pred <- predict(probit.fit, newdata = data.frame(Temp = 31), type = "response", se = TRUE)
probit.pred$fit
```

```
##         1
## 0.6963991
```

```
CI.pro.lower <- probit.pred$fit - qnorm(p=c(1 - alpha/2))*probit.pred$se
CI.pro.upper <- probit.pred$fit + qnorm(p=c(1 - alpha/2))*probit.pred$se
CI.pro <- cbind(CI.pro.lower, CI.pro.upper)
CI.pro
```

```
##   CI.pro.lower CI.pro.upper
## 1    0.1086757     1.284122
```

The predicted failure rate when temp = 31 is 0.696. We expected 95% of all similarly constructed intervals that are [0.109, 1.284] to contain the probability of failure when Temp = 31.

```
##cloglog
log.pred <- predict(log.fit, data.frame(Temp = 31), type = "response", se = TRUE)
ci.log.lower <- log.pred$fit - qnorm(p=c(1-alpha/2))*log.pred$se
ci.log.upper <- log.pred$fit + qnorm(p=c(1-alpha/2))*log.pred$se
ci.log <- cbind(ci.log.lower, ci.log.upper)
log.pred$fit
```

```
##         1
## 0.9726399
```

```
ci.log
```

```
##   ci.log.lower ci.log.upper
## 1    0.6915717     1.253708
```

The predicted failure rate when temp = 31 is 0.973. We expected 95% of all similarly constructed intervals that are [0.692, 1.254] to contain the probability of failure when Temp = 31.

```
##Logistic
pred <- predict(fit, data.frame(Temp = 31), type = "response", se = TRUE)
lower<- pred$fit - qnorm(p=c(1-alpha/2))*pred$se
upper <- pred$fit + qnorm(p=c(1-alpha/2))*pred$se
ci <- cbind(lower, upper)
pred$fit
```

```
##         1
## 0.8177744
```

```
ci
```

```
##      lower    upper
## 1 0.346496 1.289053
```

The predicted failure rate when temp = 31 is 0.818. We expected 95% of all similarly constructed intervals that are [0.346, 1.289] to contain the probability of failure when Temp = 31.

e.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
rbind(CI.pro, ci.log, ci) %>% cbind(rbind(probit.pred$fit, log.pred$fit, pred$fit)) -> compare
rownames(compare) <- c("probit", "cloglog", "logistic")
colnames(compare)[3] <- "predict"
compare
```

```
##          CI.pro.lower CI.pro.upper   predict
## probit      0.1086757     1.284122 0.6963991
## cloglog     0.6915717     1.253708 0.9726399
## logistic    0.3464960     1.289053 0.8177744
```

CIs are not similar, as the logistic model prediction has a wider CI, and probit gives a quite narrow CI. And predictied failure rates are not similar among methods.