Applied Logistic Regression – Assignment 2

Juho Ruohonen

April xx, 2017

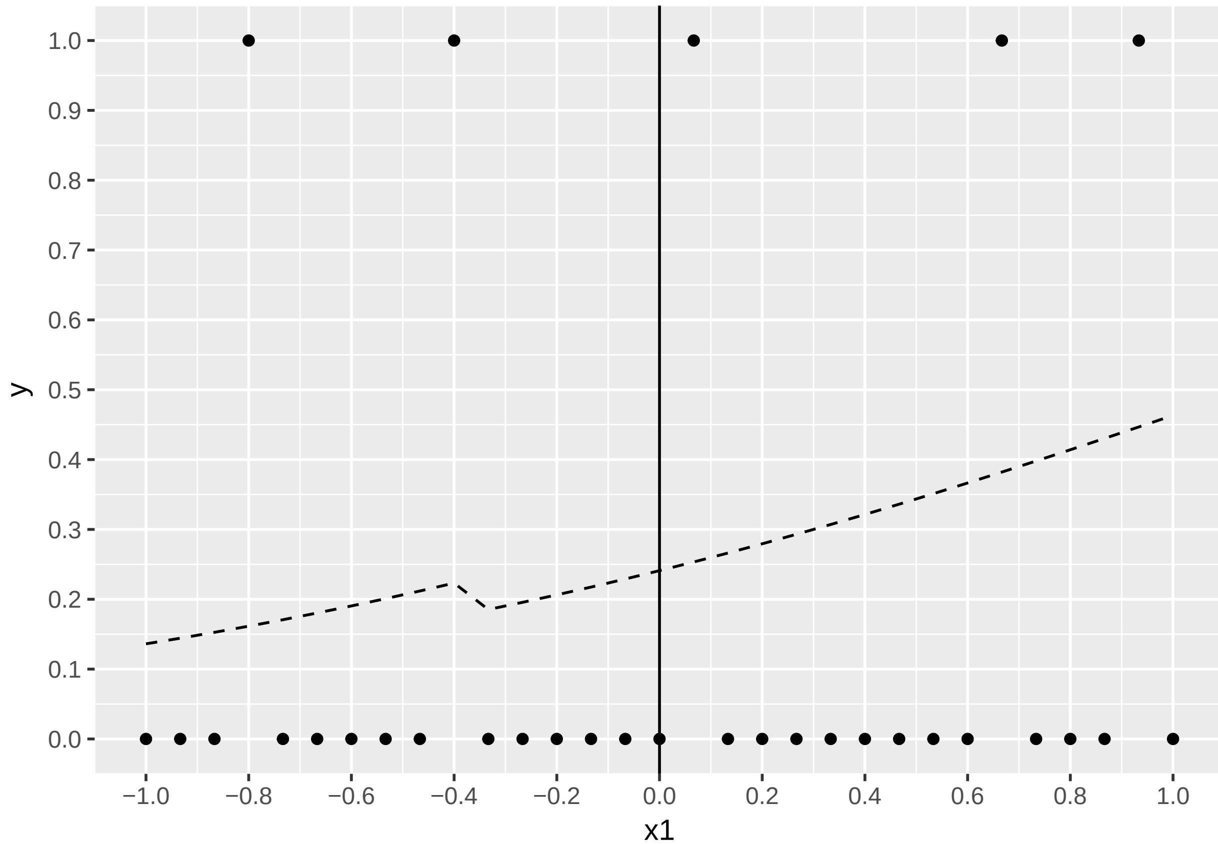
Assignment 2

1a: Data Generation

```
N<-15
n<-2*N+1
x1<-c(-N:N)/N
x2<-c(rep(0,round(n/3)),rep(1,n-round(n/3)))
x0<-rep(1,n)
X<-cbind(x0,x1,x2)
B<-c(log(0.3/0.7),1,-0.3)
lp<-X%*%B
p<-exp(lp)/(1+exp(lp))
set.seed(100) #(This is to make sure that the randomly generated outcomes
  #are the same every time the code is run)
U<-runif(n)
y<-as.numeric(U<p)
```

1b: Plotting the True Probabilities

```
library(ggplot2)
the.data<-data.frame(x0,x1,x2,lp,p,y)
ggplot(data=the.data, aes(x=x1, y=y)) +
  scale_y_continuous(limits=c(0,1), breaks=seq(from=0,to=1,by=0.1)) +
  scale_x_continuous(breaks=seq(from=-1,to=1,by=0.2)) +
  geom_point(aes(y=y,x=x1)) +
  geom_line(aes(y=p,x=x1),linetype="dashed") +
  geom_vline(xintercept = 0)
```



2: Newton's Method

```
#I think the idea is to start with the grand mean (aka intercept) of the true probability:
p1<-sum(y)/n
#Since this is logistic regression, we use the log-odds (aka logit) of the probability
  #rather than the probability itself. The other predictor coefficients are
  #are apparently given an initial value of 0:
B0<-c(log(p1/(1-p1)), 0, 0)
names(B0) <- c("x0","x1","x2")
```

So here is our starting point:

```
B0
```

```
##        x0        x1        x2
## -1.648659  0.000000  0.000000
```

Fisher Scoring Iteration 1:

```
lp0<-X%*%B0
p0<-exp(lp0)/(1+exp(lp0))
dl<-t(X)%*%(y-p0)
```

```r
w<-as.vector(p0*(1-p0))
d2l<--t(X)%*%diag(w)%*%X
B1<-B0-solve(d2l,dl)
t(B1)
```

```
##              x0       x1       x2
## [1,] -0.1788684 1.690909 -2.16969
```

The values changed, so we'll do an Iteration 2:

```r
B0<-B1
lp0<-X%*%B0
p0<-exp(lp0)/(1+exp(lp0))
dl<-t(X)%*%(y-p0)
w<-as.vector(p0*(1-p0))
d2l<--t(X)%*%diag(w)%*%X
B1<-B0-solve(d2l,dl)
t(B1)
```

```
##               x0       x1        x2
## [1,] -0.06288461 1.947025 -2.575052
```

The values changed again, so we'll do an Iteration 3:

```r
B0<-B1
lp0<-X%*%B0
p0<-exp(lp0)/(1+exp(lp0))
dl<-t(X)%*%(y-p0)
w<-as.vector(p0*(1-p0))
d2l<--t(X)%*%diag(w)%*%X
B1<-B0-solve(d2l,dl)
t(B1)
```

```
##               x0       x1        x2
## [1,] -0.03635197 1.990107 -2.638811
```

The values changed again, so we'll do an Iteration 4:

```r
B0<-B1
lp0<-X%*%B0
p0<-exp(lp0)/(1+exp(lp0))
dl<-t(X)%*%(y-p0)
w<-as.vector(p0*(1-p0))
d2l<--t(X)%*%diag(w)%*%X
B1<-B0-solve(d2l,dl)
t(B1)
```

```
##               x0       x1        x2
## [1,] -0.03595209 1.990749 -2.639743
```

The values changed very little now. But I guess we'll do an Iteration 5:

```
B0<-B1
lp0<-X%*%B0
p0<-exp(lp0)/(1+exp(lp0))
dl<-t(X)%*%(y-p0)
w<-as.vector(p0*(1-p0))
d2l<--t(X)%*%diag(w)%*%X
B1<-B0-solve(d2l,dl)
t(B1)
```

```
##               x0       x1        x2
## [1,] -0.03595201 1.990749 -2.639743
```

Only x0 changed. I suppose we'll keep going until nothing changes. Iteration 6:

```
B0<-B1
lp0<-X%*%B0
p0<-exp(lp0)/(1+exp(lp0))
dl<-t(X)%*%(y-p0)
w<-as.vector(p0*(1-p0))
d2l<--t(X)%*%diag(w)%*%X
B1<-B0-solve(d2l,dl)
t(B1)
```

```
##               x0       x1        x2
## [1,] -0.03595201 1.990749 -2.639743
```

Nothing changed. This might be what stats people call "convergence". Thus, a total of 5 Fisher Scoring Iterations were needed.

3: R's *glm()* Function

Now we'll have R's **glm()** function carry out the same procedure, and hope for the same results:

```
Y<-cbind(y,1-y)
logreg<-glm(Y~x1+x2,family=binomial(link="logit"))
summary(logreg)
```

```
##
## Call:
## glm(formula = Y ~ x1 + x2, family = binomial(link = "logit"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9036  -0.6575  -0.5010  -0.3108   2.2885
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.03595    1.35349  -0.027    0.979
## x1           1.99075    1.66216   1.198    0.231
## x2          -2.63974    2.24718  -1.175    0.240
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27.392  on 30  degrees of freedom
## Residual deviance: 25.588  on 28  degrees of freedom
## AIC: 31.588
##
## Number of Fisher Scoring iterations: 5
```

Indeed, the coefficients are the same (though the model summary rounds them). The number of Fisher Scoring iterations performed is also identical. This suggests major mistakes have not been made.

4: Variance-Covariance Matrices

```r
#Here's the variance-covariance matrix of the manually computed model:
(V<--solve(d2l))
```

```
##           x0        x1        x2
## x0  1.831951  1.815212 -2.821587
## x1  1.815212  2.762775 -3.321450
## x2 -2.821587 -3.321450  5.049817
```

```r
#Here's the variance-covariance matrix of the model computed by the glm() function:
vcov(logreg)
```

```
##             (Intercept)        x1        x2
## (Intercept)    1.831948  1.815206 -2.821579
## x1             1.815206  2.762766 -3.321437
## x2            -2.821579 -3.321437  5.049798
```

Everything is identical again, notwithstanding rounding.

5: Standard Errors of Model Coefficients:

```r
#Predictor coefficient standard errors in the manually computed model:
sqrt(diag(V))
```

```
##       x0       x1       x2
## 1.353496 1.662160 2.247180
```

```r
#Predictor coefficient standard errors in glm()-computed model:
sqrt(diag(vcov(logreg)))
```

```
## (Intercept)          x1          x2
##    1.353495    1.662157    2.247176
```

6: x1 as the Only Predictor

```r
logreg.x1<-glm(Y~x1, family=binomial(link="logit"))
summary(logreg.x1)
```

```
##
## Call:
```

```
## glm(formula = Y ~ x1, family = binomial(link = "logit"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6803  -0.6218  -0.5730  -0.5250   2.0250
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.6606     0.4934  -3.365 0.000765 ***
## x1            0.3149     0.8264   0.381 0.703191
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27.392  on 30  degrees of freedom
## Residual deviance: 27.245  on 29  degrees of freedom
## AIC: 31.245
##
## Number of Fisher Scoring iterations: 4
```

7: x2 as the Only Predictor

```
logreg.x2<-glm(Y~x2, family=binomial(link="logit"))
summary(logreg.x2)
```

```
##
## Call:
## glm(formula = Y ~ x2, family = binomial(link = "logit"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6681  -0.6117  -0.5553  -0.5553   1.9728
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.3863     0.7906  -1.754   0.0795 .
## x2           -0.4055     1.0069  -0.403   0.6872
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27.392  on 30  degrees of freedom
## Residual deviance: 27.233  on 29  degrees of freedom
## AIC: 31.233
##
## Number of Fisher Scoring iterations: 4
```

8: Interpreting the Results

So, the full model (with both predictors included) reports a positive effect on the outcome for **x1** and a negative one for **x2**, both of them statistically non-significant. A very similar result is obtained when we only include **x1** or **x2** in the model – the former has a positive coefficient, the latter a negative one. My interpretation is that increasing magnitude of the continuous property **x1** favors the occurrence of the outcome, while membership in group **x2** (represented by value 1 of the dichotomous predictor) disfavors it. This is well illustrated by our graph in exercise 1 – the slope is consistently ascending as a function of **x1**. The single downward blip in the graph (at about x1 = 0.3) occurs when Group Identity (x2) changes from 0 to 1.