Binary Logistic Regression DrPH (Epidemiology)

Kamarul Imran M

1 February 2016

Contents

Introduction

Modeling Binary Outcome Data

The suggested steps are:

1. Describe data
2. Explore data - Exploratory Data Analysis
3. Estimate parameters
4. Interpret parameters
5. Make Inference
6. Calculate the fitted values
7. Make Prediction
8. Assessing model
9. Remedy model

Prepare workspace

Locate files

- Browse your folders.
- Look for the files.
- Check the path to the folder containing the files

Set the folder

Set our working directory. REMEMBER! your working directory (working folder) is different from my working directory

```
# this is my working directory. You have to specify yours
setwd("E:/Epi_Stat_Matters/LectureNotes2015/binary-logistic/binary-logistic-DrPH-2015/BinaryLogisticDrPH
```

Read data

- Read our data in the working folder
- Then, save as a csv file in our working directory

```
mydata <- read.csv('datalogistic.csv', sep = ",", header = TRUE)
```

Preliminary analysis

Describe your data

```
# observe data the first 10 observations
head(mydata,10)
```

More fancy, we can use *psych::describe* function

```
library(psych)
describe(mydata)
```

```
##       vars   n   mean     sd median trimmed    mad    min max  range  skew
## ID       1 400 200.50 115.61  200.5  200.50 148.26   1.00 400 399.00  0.00
## admit    2 400   0.32   0.47    0.0    0.27   0.00   0.00   1   1.00  0.78
## gre      3 400 587.70 115.52  580.0  589.06 118.61 220.00 800 580.00 -0.14
## gpa      4 400   3.39   0.38    3.4    3.40   0.40   2.26   4   1.74 -0.21
## rank     5 400   2.48   0.94    2.0    2.48   1.48   1.00   4   3.00  0.10
##       kurtosis   se
## ID       -1.21 5.78
## admit    -1.39 0.02
## gre      -0.36 5.78
## gpa      -0.60 0.02
## rank     -0.91 0.05
```

Admit and Rank are taken as numerical variable which does not make sense. We need to convert them to categorical (factor) variables.

```
str(mydata$admit)
```

```
##  int [1:400] 0 1 1 1 0 1 1 0 1 0 ...
```

```
str(mydata$rank)
```

```
##  int [1:400] 3 3 1 4 4 2 1 2 3 2 ...
```

```
mydata$admit2 <- factor(mydata$admit, labels = c('no','yes'))
mydata$rank2 <- factor(mydata$rank, labels = c('first', 'second', 'third', 'fourth'))
head(mydata)
```

```
##   ID admit gre  gpa rank admit2  rank2
## 1  1     0 380 3.61    3     no  third
## 2  2     1 660 3.67    3    yes  third
## 3  3     1 800 4.00    1    yes  first
## 4  4     1 640 3.19    4    yes fourth
## 5  5     0 520 2.93    4     no fourth
## 6  6     1 760 3.00    2    yes second
```

```
str(mydata)
```

```
## 'data.frame':    400 obs. of  7 variables:
##  $ ID    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ admit : int  0 1 1 1 0 1 1 0 1 0 ...
##  $ gre   : int  380 660 800 640 520 760 560 400 540 700 ...
##  $ gpa   : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
##  $ rank  : int  3 3 1 4 4 2 1 2 3 2 ...
##  $ admit2: Factor w/ 2 levels "no","yes": 1 2 2 2 1 2 2 1 2 1 ...
##  $ rank2 : Factor w/ 4 levels "first","second",..: 3 3 1 4 4 2 1 2 3 2 ...
```

Explore your data

Use plots like * Histogram for numerical variables * and barplot for categorical variables, at least.

Model parameters

Estimate parameters

- we estimate the logit or the log odds using `glm` function.
- We used **summary** to see the results stored as the glm model for example **mylogit**
- We used **coefficients** to examine the regression coefficients

```
mylogit <- glm(admit2 ~ gre + gpa + rank2,family = 'binomial'(link = logit),data=mydata)
summary(mylogit)
```

```
##
## Call:
## glm(formula = admit2 ~ gre + gpa + rank2, family = binomial(link = logit),
##     data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2second -0.675443   0.316490  -2.134 0.032829 *
## rank2third  -1.340204   0.345306  -3.881 0.000104 ***
## rank2fourth -1.551464   0.417832  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

If we would like to see the estimated *beta* then we can use this

```
coefficients(mylogit)
```

```
##  (Intercept)          gre          gpa  rank2second   rank2third
## -3.989979073  0.002264426  0.804037549 -0.675442928 -1.340203916
##  rank2fourth
## -1.551463677
```

To obtain the odds ratios and their 95% CI, we need to exponentiate using *exp* the regression coefficients or the *betas*

```
exp(coefficients(mylogit))
```

```
## (Intercept)         gre         gpa rank2second  rank2third rank2fourth
##   0.0185001   1.0022670   2.2345448   0.5089310   0.2617923   0.2119375
```

```
exp(confint(mylogit))
```

```
## Waiting for profiling to be done...
```

```
##                    2.5 %    97.5 %
## (Intercept) 0.001889165 0.1665354
## gre         1.000137602 1.0044457
## gpa         1.173858216 4.3238349
## rank2second 0.272289674 0.9448343
## rank2third  0.131641717 0.5115181
## rank2fourth 0.090715546 0.4706961
```

Interpret parameters

1. Interpret the *betas* (the log odds) and their 95% CIs
2. Interpret the *odds ratios* and their 95% CIs

Inference

Here, we examine

1. the p-values (hypothesis testing) and
2. the confidence intervals.

What is the p-values?

The p-value is defined as the probability of obtaining a result equal to or "more extreme" than what was actually observed, when the null hypothesis is true. In frequentist inference, the p-value is widely used in statistical hypothesis testing, specifically in null hypothesis significance testing. Ref: https://en.wikipedia.org/wiki/P-value

For example, suppose that a vaccine study produced a P value of 0.04. This P value indicates that if the vaccine had no effect, you'd obtain the observed difference or more in 4% of studies due to random sampling error. Reg: http://blog.minitab.com/blog/adventures-in-statistics-2/how-to-correctly-interpret-p-values

- First, using the method of maximum likelihood
- Next, using the SE method (function **confint.default**)

```
confint(mylogit)
```

```
## Waiting for profiling to be done...
```

```
##                     2.5 %       97.5 %
## (Intercept) -6.2716202334 -1.792547080
## gre          0.0001375921  0.004435874
## gpa          0.1602959439  1.464142727
## rank2second -1.3008888002 -0.056745722
## rank2third  -2.0276713127 -0.670372346
## rank2fourth -2.4000265384 -0.753542605
```

```
confint.default(mylogit)
```

```
##                     2.5 %       97.5 %
## (Intercept) -6.2242418514 -1.755716295
## gre          0.0001202298  0.004408622
## gpa          0.1536836760  1.454391423
## rank2second -1.2957512650 -0.055134591
```

```
## rank2third  -2.0169920597 -0.663415773
## rank2fourth -2.3703986294 -0.732528724
```

Fitted and predicted values

Fitted values

The fitted values are the expected values of the model. These expected values are the predicted probability for each observation (each patient) in the dataset. You can use 2 functions of getting the fitted values

```
fitted(mylogit)
predict(mylogit, type = 'response')
```

Manually, we can do this to calculate (verify) the conditional probability of being admitted for the current dataset

```
head(fitted(mylogit))
```

```
##         1         2         3         4         5         6
## 0.1726265 0.2921750 0.7384082 0.1783846 0.1183539 0.3699699
```

```
# calculate the logistic probability for the 1st observation
exp(-1.567)/(1+exp(-1.567))
```

```
## [1] 0.1726445
```

Predicted values

In R, you can prediction of the outcome based on a set of a new data

Create a new data to make prediction

Similarly, one of the important objectives in modelling is to perform prediction based on the model using new data.

We can perform these predictions:

1. Predict the log odds for having the outcome
2. Predict the conditional probability for having the outcome

Let us say we have these data

gre = 380 gpa = 3.61 rank = first,second, third, fourth

First, we create a data frame

```
new_datal <- data.frame( gre = 380, gpa = 3.61, rank2 = c('first','second','third', 'fourth'))
new_datal
```

```
##   gre  gpa  rank2
## 1 380 3.61  first
## 2 380 3.61 second
## 3 380 3.61  third
## 4 380 3.61 fourth
```

Now, we predict the log odds for being **admitted** for a population with **gre=30**, **gpa=3.61** and for different **rank**

```
pred.logit<-predict(mylogit,newdata = new_datal, type='link')
pred.logit
```

```
##          1          2          3          4
## -0.2269217 -0.9023646 -1.5671256 -1.7783854
```

We can confirm this by calculate this (using first and third rank)

```
-3.99 + 0.00226*380 + 0.8041*3.61 + 0
```

```
## [1] -0.228399
```

```
-3.99 + 0.00226*380 + 0.8041*3.61 - 1.34
```

```
## [1] -1.568399
```

Notice, that similarity between **predict(x, type='response')** and **fitted** Remember, we can calculate the conditional probability of having the outcome

Compare models

We compare a model with **gre** and without **gre**. This is done using the deviance method

```
mylogit2 <- glm(admit2 ~ gpa + rank2, family = 'binomial'(link = logit),data=mydata)
anova(mylogit, mylogit2, test = 'Chisq')
```

```
## Analysis of Deviance Table
##
## Model 1: admit2 ~ gre + gpa + rank2
## Model 2: admit2 ~ gpa + rank2
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       394     458.52
## 2       395     462.88 -1  -4.3578  0.03684 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value shows that mylogit and mylogit2 are different. It suggests the importance of **gre** at the level of significance of 5%. So should we keep **gre**? Perhaps yes, if we take the p-value as the requirement to assess for variable significance.

Model assessment

Linearity in logits

gre is tested for linearity in logit. **gre** is linear but it is rescaled to produce less decimals

The linearity of logits is tested using library *mfp* package

```
library(mfp)
```

```
## Loading required package: survival
```

```
mylogit3 <- mfp(admit2 ~ fp(gre)+ gpa + rank2, family = 'binomial'(link = logit),data=mydata,verbose=T)
```

```
##
##  Variable    Deviance    Power(s)
## ------------------------------------------------
## Cycle 1
##  rank2second
##              463.096
##              458.517     1
##
##
##
##  rank2third
##              474.043
##              458.517     1
##
##
##
##  rank2fourth
##              473.551
##              458.517     1
##
##
##
##  gpa
##              464.532
##              458.517     1
##
##
##
##  gre
##              462.875
##              458.517     1
##              458.415     0
##              458.366     -2 -2
##
##
## Tansformation
##             shift scale
## rank2second     0     1
## rank2third      0     1
## rank2fourth     0     1
## gpa             0     1
## gre             0  1000
##
## Fractional polynomials
##             df.initial select alpha df.final power1 power2
## rank2second          1      1  0.05        1      1      .
## rank2third           1      1  0.05        1      1      .
## rank2fourth          1      1  0.05        1      1      .
## gpa                  1      1  0.05        1      1      .
## gre                  4      1  0.05        1      1      .
##
##
## Transformations of covariates:
##               formula
```

```
## gre   I((gre/1000)^1)
## gpa               gpa
## rank2           rank2
##
##
## Deviance table:
##           Resid. Dev
## Null model    499.9765
## Linear model  458.5175
## Final model   458.5175
```

Now, let us check the estimated parameters based on fractional polynomials

```
summary(mylogit3)
```

```
##
## Call:
## glm(formula = admit2 ~ rank2 + gpa + I((gre/1000)^1), family = binomial(link = logit),
##     data = mydata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -3.9900     1.1400  -3.500 0.000465 ***
## rank2second      -0.6754     0.3165  -2.134 0.032829 *
## rank2third       -1.3402     0.3453  -3.881 0.000104 ***
## rank2fourth      -1.5515     0.4178  -3.713 0.000205 ***
## gpa               0.8040     0.3318   2.423 0.015388 *
## I((gre/1000)^1)   2.2644     1.0940   2.070 0.038465 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

```
mylogit3$fptable
```

```
##             df.initial select alpha df.final power1 power2
## rank2second          1      1  0.05        1      1      .
## rank2third           1      1  0.05        1      1      .
## rank2fourth          1      1  0.05        1      1      .
## gpa                  1      1  0.05        1      1      .
## gre                  4      1  0.05        1      1      .
```

Overall model fitness

To do these diagnostics, you can use *LogisticDx* package. This package produces the diagnostic measures for a binary regression model based on covariate pattern

This package produces the Goodness-of-fit for binomial regression including the Hosmer-Lemeshow GOF test and the ROC curve. Usually, the number of groups (quantiles) equal 10 to perform the Hosmer-Lemeshow test. At the same time, we plot the ROC curve. Similarly, we can check the *LogisticDx::auc* value

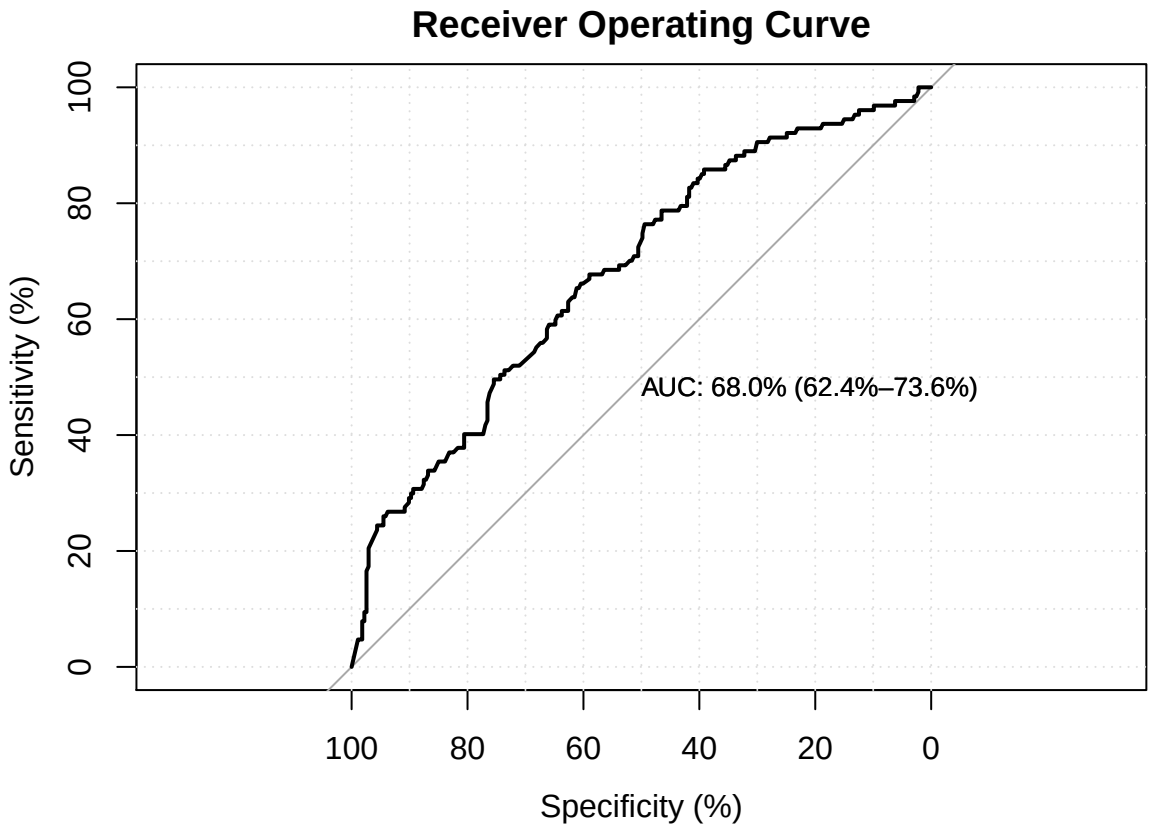
The *LogisticDx::dx* produces the estimates of residual diagnostics such as 1. standardized residuals 2. dchi-square 3. ddeviance

```
library('LogisticDx')
dx_mylogit2 <- dx(mylogit2,byCov=T)
head(dx_mylogit2, 10)
```

```
##     (Intercept)  gpa rank2second rank2third rank2fourth y         P n
##  1:           1 3.94           1          0           0 1 0.5001242 2
##  2:           1 3.30           1          0           0 1 0.3378663 3
##  3:           1 3.99           0          1           0 1 0.3412259 3
##  4:           1 3.35           1          0           0 1 0.3497327 3
##  5:           1 3.17           1          0           0 1 0.3079792 3
##  6:           1 4.00           0          0           0 6 0.6779993 9
##  7:           1 4.00           1          0           0 4 0.5158996 8
##  8:           1 3.78           1          0           0 1 0.4581403 2
##  9:           1 3.75           1          0           0 1 0.4503162 2
## 10:           1 3.05           1          0           0 1 0.2817434 3
##          yhat            Pr            dr           h           sPr
##  1: 1.0002483 -0.0003511985 -0.0003511985 0.014377039 -0.0003537507
##  2: 1.0135990 -0.0165997514 -0.0166181074 0.006902268 -0.0166573376
##  3: 1.0236777 -0.0288329487 -0.0288874508 0.015618469 -0.0290607843
##  4: 1.0491980 -0.0595625368 -0.0597856465 0.006844794 -0.0597674360
##  5: 0.9239375  0.0951239167  0.0944359041 0.007474218  0.0954814103
##  6: 6.1019933 -0.0727624780 -0.0725435839 0.021459082 -0.0735559800
##  7: 4.1271968 -0.0899872045 -0.0899644454 0.015995728 -0.0907156608
##  8: 0.9162805  0.1188143108  0.1186054483 0.010787822  0.1194604169
##  9: 0.9006323  0.1412260677  0.1408758662 0.010240239  0.1419547634
## 10: 0.8452301  0.1986365068  0.1953092157 0.008468306  0.1994829438
##               sdr       dChisq         dDev        dBhat
##  1: -0.0003537506 1.251395e-07 1.251395e-07 1.825379e-09
##  2: -0.0166757573 2.774669e-04 2.780809e-04 1.928462e-06
##  3: -0.0291157171 8.445292e-04 8.477250e-04 1.339953e-05
##  4: -0.0599913131 3.572146e-03 3.598958e-03 2.461912e-05
##  5:  0.0947908121 9.116700e-03 8.985298e-03 6.865333e-05
##  6: -0.0733346987 5.410482e-03 5.377978e-03 1.186501e-04
##  7: -0.0906927175 8.229331e-03 8.225169e-03 1.337740e-04
##  8:  0.1192504186 1.427079e-02 1.422066e-02 1.556297e-04
##  9:  0.1416027549 2.015115e-02 2.005134e-02 2.084876e-04
## 10:  0.1961414744 3.979344e-02 3.847148e-02 3.398611e-04
```

Use *LogisticDx::gof* to produce 1. Hosmer-Lemeshow GOF test 2. Osius and Rojek's tests 3. the auc

```
fit.mylogit2 <- gof(mylogit2, g=10, plotROC = T)
```




```
fit.mylogit2
```

```
##       chiSq  df     pVal
## PrI  400.78 395 0.409645
## drI  462.88 395 0.010379 *
## PrG  255.36 255 0.481870
## drG  294.60 255 0.044609 *
## PrCT 255.36 255 0.481870
## drCT 294.60 255 0.044609 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##                       val df     pVal
## HL chiSq         8.190458  8 0.415090
## mHL F            1.484491  9 0.153918
## OsRo Z           0.027903 NA 0.977740
## SstPgeq0.5 Z     0.217564 NA 0.827769
## SstPl0.5 Z       0.422859 NA 0.672398
## SstBoth chiSq    0.226144  2 0.893086
## SllPgeq0.5 chiSq 0.047739  1 0.827045
## SllPl0.5 chiSq   0.176745  1 0.674186
## SllBoth chiSq    0.177398  2 0.915121
```

To obtain the contigency table for the Hosmer-Lemeshow GOF test, we can use

```
fit.mylogit2$ctHL
```

```
##         P y1     y1hat y0    y0hat  n      Pbar
##  1: 0.149  6  5.037833 34 34.96217 40 0.1259458
##  2: 0.192  5  6.824095 35 33.17591 40 0.1706024
##  3: 0.219  7  8.260683 33 31.73932 40 0.2065171
##  4: 0.258 11  9.327547 28 29.67245 39 0.2391679
##  5: 0.293 12 11.338710 29 29.66129 41 0.2765539
##  6:  0.34 15 12.556219 25 27.44378 40 0.3139055
##  7: 0.376 14 13.796927 25 25.20307 39 0.3537674
##  8: 0.437 14 16.561867 27 24.43813 41 0.4039480
##  9: 0.511 13 18.253216 25 19.74678 38 0.4803478
## 10: 0.678 30 25.042903 12 16.95710 42 0.5962596
```

and for the GOF test

```
fit.mylogit2$gof
```

```
##          test  stat        val df      pVal
## 1:         HL chiSq 8.19045835  8 0.4150903
## 2:        mHL     F 1.48449073  9 0.1539182
## 3:       OsRo     Z 0.02790263 NA 0.9777398
## 4: SstPgeq0.5     Z 0.21756392 NA 0.8277689
## 5:   SstPl0.5     Z 0.42285933 NA 0.6723979
## 6:    SstBoth chiSq 0.22614408  2 0.8930863
## 7: SllPgeq0.5 chiSq 0.04773921  1 0.8270450
## 8:   SllPl0.5 chiSq 0.17674490  1 0.6741857
## 9:    SllBoth chiSq 0.17739848  2 0.9151208
```

We can also perform model fitness by using *ROCR* package

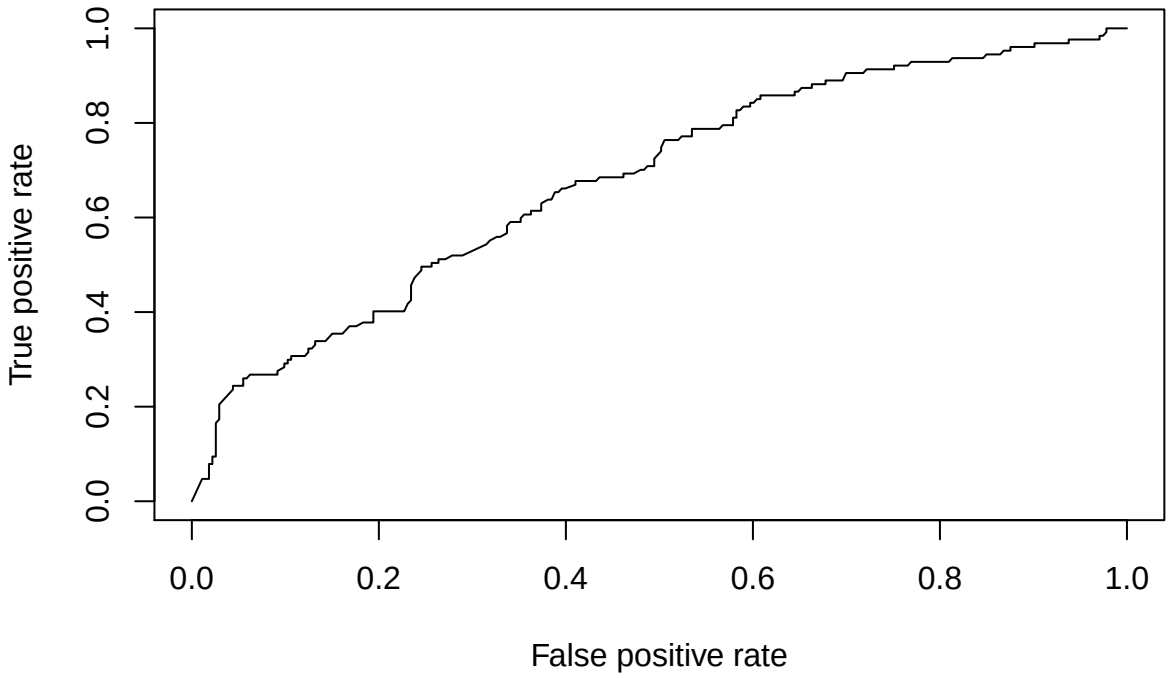
```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
pred.prob2 <- predict(mylogit2, type='response')
pred.prob22 <- prediction(pred.prob2, mydata$admit2)
pred.prob22f <- performance(pred.prob22, measure='tpr', x.measure='fpr')
plot(pred.prob22f)
```



Using *ROCR* package, we can also calculate the *AUC*

```
auc2<-performance(pred.prob22, measure='auc')
auc2@y.values[[1]]
```

```
## [1] 0.6801073
```

Another package is *MKmisc* package, to perform the Hosmer-Lemeshow test of GOF

```
library(MKmisc)
```

```
##
## Attaching package: 'MKmisc'
```

```
## The following object is masked from 'package:psych':
##
##     corPlot
```

```
HLgof.test(fit = fitted(mylogit2), obs = mydata$admit)
```

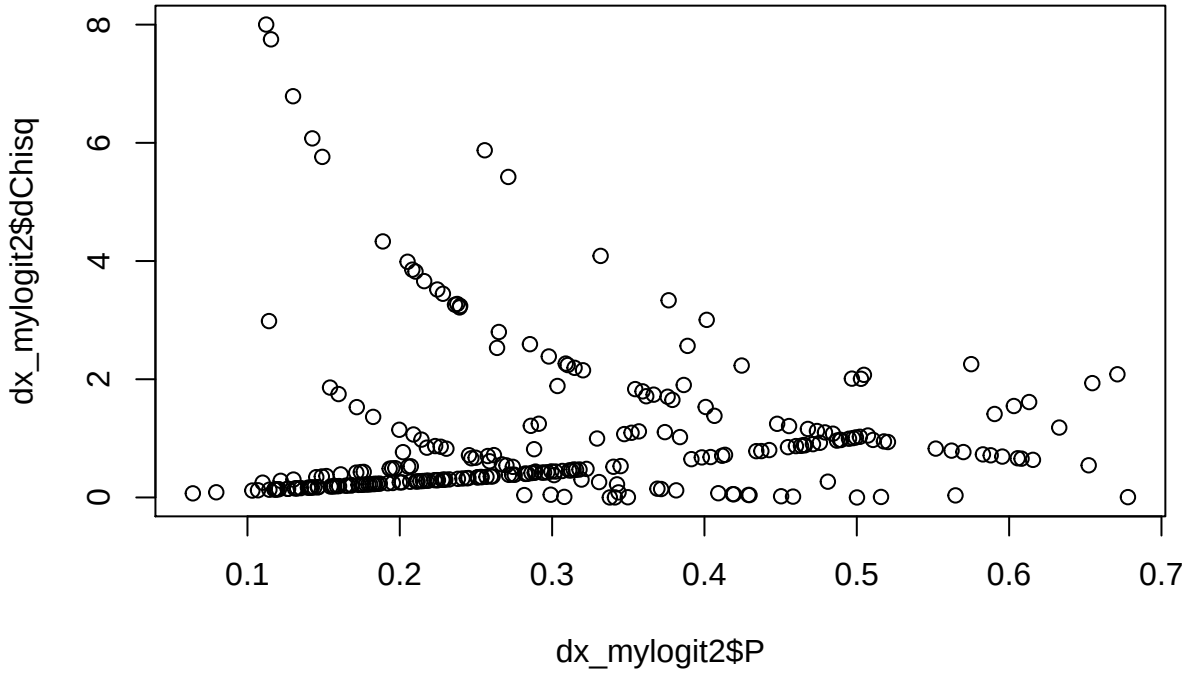
```
## $C
##
##  Hosmer-Lemeshow C statistic
##
## data:  fitted(mylogit2) and mydata$admit
## X-squared = 9.8564, df = 8, p-value = 0.2752
##
##
## $H
```

```
##
##  Hosmer-Lemeshow H statistic
##
## data:  fitted(mylogit2) and mydata$admit
## X-squared = 7.6802, df = 8, p-value = 0.4653
```

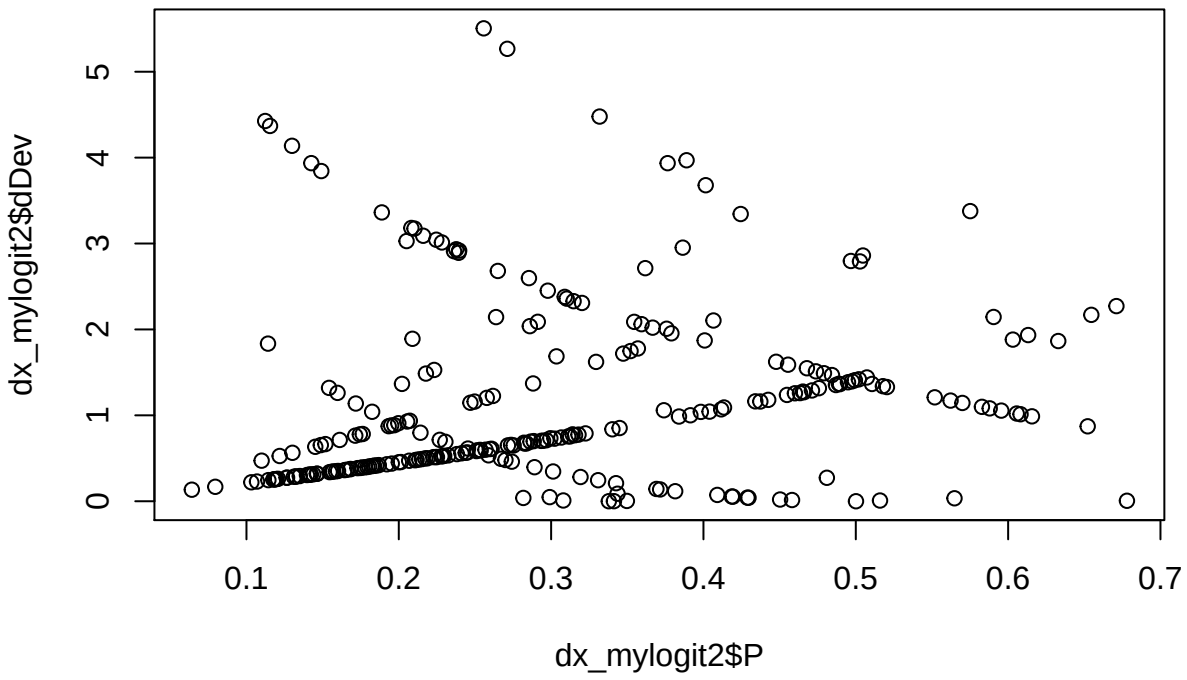
Diagnostic statistics

We can start of plotting these

```
plot(dx_mylogit2$P, dx_mylogit2$dChisq)
```




```
plot(dx_mylogit2$P, dx_mylogit2$dDev)
```



You may try and plot the covariate pattern numbers (identifiers) by using this

```
library(epiR)
```

```
## Package epiR 0.9-79 is loaded
## Type help(epi.about) for summary information
##
```

```
dat.mf2 <- model.frame(mylogit2)
head(dat.mf2)
```

```
##   admit2  gpa  rank2
## 1     no 3.61  third
## 2    yes 3.67  third
## 3    yes 4.00  first
## 4    yes 3.19 fourth
## 5     no 2.93 fourth
## 6    yes 3.00 second
```

```
cv_mf2 <- epi.cp(dat.mf2[-1])
head(cv_mf2$cov.pattern)
```

```
##   id n  gpa  rank2
## 1  1 2 3.61  third
## 2  2 2 3.67  third
## 3  3 9 4.00  first
## 4  4 2 3.19 fourth
```

```
## 5  5 2 2.93 fourth
## 6  6 2 3.00 second
```

The *plot* function Will return many diagnostic plots

```
plot(mylogit2)
```

Using K-fold validation. Will not discuss here.

Logistic model with interaction of predictors

Let use see how we deal an interaction. First, read data from this text file.

Columns (variables) no 2, and from 5 to 10 need to be converted to categorical (factor) variables. We will use *lapply* function for that.

```
data.l<-read.table("LOWBWT.txt",header=T)
data.l[,c(2,5:10)]<-lapply(data.l[,c(2,5)],factor)
```

Now, observe the first few data

```
head(data.l)
```

```
##   ID LOW AGE LWT RACE SMOKE PTL HT UI FTV  BWT
## 1  4   1  28 120    3     1   3  1  3   1  709
## 2 10   1  29 130    1     1   1  1  1   1 1021
## 3 11   1  34 187    2     1   2  1  2   1 1135
## 4 13   1  25 105    3     1   3  1  3   1 1330
## 5 15   1  25  85    3     1   3  1  3   1 1474
## 6 16   1  27 150    3     1   3  1  3   1 1588
```

To simulate a binary predictor variable, we now recode LWT to LWD (LWT<110 vs >=110)

```
data.l$LWD <- findInterval(data.l$LWT, 110)
head(data.l$LWD)
```

```
## [1] 1 1 1 0 0 1
```

Let us verify our categorization

```
data.l$LWD <- factor(data.l$LWD, labels = c("less 110",">=110"))
head(data.l$LWD, 10)
```

```
##  [1] >=110    >=110    >=110    less 110 less 110 >=110    less 110
##  [8] >=110    >=110    >=110
## Levels: less 110 >=110
```

```
head(data.l$LWT, 10)
```

```
##  [1] 120 130 187 105  85 150  97 128 132 165
```

```
str(data.l$LWD)
```

```
##  Factor w/ 2 levels "less 110",">=110": 2 2 2 1 1 2 1 2 2 2 ...
```

Model the relationship between the outcome variable (LOW = 0,1) with predictors of LWD and AGE interacting with each other. You may use # to perform that

```
mod.lwd.age <- glm(LOW ~ LWD*AGE, family = binomial(link =logit ),data=data.l)
summary(mod.lwd.age)
```

```
##
## Call:
## glm(formula = LOW ~ LWD * AGE, family = binomial(link = logit),
##     data = data.l)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4257  -0.8554  -0.6960   1.1602   2.0329
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.16959    1.46515  -0.798   0.4247
## LWD>=110      1.94409    1.72481   1.127   0.2597
## AGE           0.05262    0.06449   0.816   0.4145
## LWD>=110:AGE -0.13220    0.07570  -1.746   0.0807 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 234.67  on 188  degrees of freedom
## Residual deviance: 221.14  on 185  degrees of freedom
## AIC: 229.14
##
## Number of Fisher Scoring iterations: 4
```

Predict our model (with a two-way interaction between age and LWD using a new set of data. We can create such a dataset with this

```
newdata.2<-data.frame(AGE = c(15,15,20,20),LWD= rep(c("less 110",">=110"), 2))
newdata.2
```

```
##   AGE      LWD
## 1  15 less 110
## 2  15    >=110
## 3  20 less 110
## 4  20    >=110
```

Now let us predict the log odds for the model with the interaction term

```
predict(mod.lwd.age, newdata = newdata.2)
```

```
##          1          2          3          4
## -0.3802252 -0.4190874 -0.1171023 -0.8169483
```

Can you prove the predicted log odds manually?

```
-1.1696 + 0 + 0.0526*15 + 0
```

```
## [1] -0.3806
```

```
-1.1696 + 1.944*1 + 0.0526*15 - 0.1322*1*15
```

```
## [1] -0.4196
```

```
-1.1696 + 0 + 0.0526*20 + 0
```

```
## [1] -0.1176
```

```r
-1.1696 + 1.944*1 + 0.0526*20 - 0.1322*1*20
```

```
## [1] -0.8176
```

Resources

1. http://www.ats.ucla.edu/stat/r/dae/logit.htm
2. https://cran.r-project.org/web/packages/HSAUR/vignettes/Ch_logistic_regression_glm.pdf