# Biostats - Multiple Logistic Regression

Rob Leonard (robleonard@tamu.edu (mailto:robleonard@tamu.edu))

## Cleveland Clinic: Heart Disease Dataset

**1) Load data and categorize thal into 2 groups.**

```
origdata = read.csv("heart.csv", header=TRUE, sep = ",")   # read in original data
names(origdata)[1] = "age"
heartdata = data.frame(origdata)  # set up adjusted data set
heartdata$thal = ifelse(heartdata$thal>2,0,1)
```

**2) Scale the non binary/categorical variables.**

```
heartdata$age = scale(heartdata$age)
heartdata$trestbps = scale(heartdata$trestbps)
heartdata$chol = scale(heartdata$chol)
heartdata$thalach = scale(heartdata$thalach)
heartdata$oldpeak = scale(heartdata$oldpeak)
heartdata$ca = scale(heartdata$ca)
```

**3) Fit a logistic regression model using all 13 explanatory variables.**

```
model1 = glm(target~., data = heartdata, family="binomial")
summary(model1)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = heartdata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6557  -0.3632   0.1784   0.5674   2.5937
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.23986    0.76612  -1.618 0.105585
## age         -0.02293    0.21319  -0.108 0.914358
## sex         -1.45793    0.48590  -3.000 0.002696 **
## cp           0.80501    0.18579   4.333 1.47e-05 ***
## trestbps    -0.32653    0.18330  -1.781 0.074856 .
## chol        -0.23079    0.19619  -1.176 0.239452
## fbs          0.15650    0.52895   0.296 0.767332
## restecg      0.53150    0.35470   1.498 0.134015
## thalach      0.49085    0.23753   2.066 0.038784 *
## exang       -0.90543    0.41334  -2.191 0.028488 *
## oldpeak     -0.60350    0.25294  -2.386 0.017038 *
## slope        0.50741    0.35706   1.421 0.155287
## ca          -0.80236    0.19945  -4.023 5.75e-05 ***
## thal         1.40201    0.38008   3.689 0.000225 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 417.64  on 302  degrees of freedom
## Residual deviance: 207.21  on 289  degrees of freedom
## AIC: 235.21
##
## Number of Fisher Scoring iterations: 6
```

### 4) Predict probability of disease for 3 values and determine 95% CI’s.

```
page = c(68,75,78)
psex = c(1,0,1)
pcp = c(3,3,0)
ptrestbps = c(145,145,144)
pchol = c(233,150,193)
pfbs = c(1,1,1)
prestecg = c(0,0,1)
pthalach = c(150,150,90)
pexang = c(0,0,0)
poldpeak = c(2.3,2.3,3.4)
pslope = c(0,0,1)
pca = c(0,0,2)
pthal = c(0,1,0)
pdata = data.frame(page,psex,pcp,ptrestbps,pchol,pfbs,prestecg,pthalach,pexang,poldpeak,pslope,p
ca,pthal)
names(pdata)[1:13] = names(origdata)[1:13]
# scale the data
pdata$age = scale(pdata$age, center = mean(origdata$age), scale = sd(origdata$age))
pdata$trestbps = scale(pdata$trestbps, center = mean(origdata$trestbps), scale = sd(origdata$tre
stbps))
pdata$chol = scale(pdata$chol, center = mean(origdata$chol), scale = sd(origdata$chol))
pdata$thalach = scale(pdata$thalach, center = mean(origdata$thalach), scale = sd(origdata$thalac
h))
pdata$oldpeak = scale(pdata$oldpeak, center = mean(origdata$oldpeak), scale = sd(origdata$oldpea
k))
pdata$ca = scale(pdata$ca, center = mean(origdata$ca), scale = sd(origdata$ca))
# predict
pout = predict.glm(model1, newdata = pdata, se.fit=TRUE)
1/(1+exp(-as.numeric(pout[[1]])))
```

```
## [1] 0.395098688 0.941910779 0.006322255
```

```
# confidence intervals
ll=1/(1+exp(-(as.numeric(pout[[1]])-1.96*as.numeric(pout[[2]]))))
ll
```

```
## [1] 0.1297430515 0.6868298595 0.0007611971
```

```
ul=1/(1+exp(-(as.numeric(pout[[1]])+1.96*as.numeric(pout[[2]]))))
ul
```

```
## [1] 0.74103848 0.99172758 0.05045901
```

| Observation | Estimated Probability have Disease | 95% Confidence Interval |
|---|---|---|
| 1 | 0.395 | ( 0.129,0.741) |
| 2 | 0.942 | ( 0.687, 0.992) |
| 3 | 0.006 | ( 0.0008, 0.050) |

**5) Check the adequecy of the model using the Hosmer-Lemeshow test.** The null hypothesis $H_0$ : is that the model fits the data well, versus the alternative hypothesis $H_a$ : that the model does not fit the data well.

```
library(generalhoslem)
```

```
## Loading required package: reshape
```

```
## Loading required package: MASS
```

```
logitgof(heartdata$target, fitted(model1))
```

```
## Warning in logitgof(heartdata$target, fitted(model1)): At least one cell in the
## expected frequencies table is < 1. Chi-square approximation may be incorrect.
```

```
##
##  Hosmer and Lemeshow test (binary model)
##
## data:  heartdata$target, fitted(model1)
## X-squared = 8.8059, df = 8, p-value = 0.3589
```

The test statistic is 8.8059, using an approximate chi-squared distribution with 10 groups and 8 degrees of freedom. The p-value is high at 0.359, so we do not have significant evidence to reject the null hypothesis that the model fits the data well.
However, at least one cell has an expected frequency of < 1 so the approximation may be incorrect. If we use a g=5, the p-value is even higher, however with small g's, the HL test may not be able to detect a model mispecification.

**6) Rerun the model using only the first and last 100 observations.**

```
smallerdata = heartdata[c(1:100,204:303),]
model1red = glm(target~., data = smallerdata, family="binomial")
summary(model1red)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = smallerdata)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.79782  -0.32264   0.03937   0.47638   2.64926
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.89721    1.01561  -1.868 0.061755 .
## age          0.02639    0.28838   0.092 0.927078
## sex         -1.21318    0.64120  -1.892 0.058484 .
## cp           0.72699    0.22712   3.201 0.001370 **
## trestbps    -0.71427    0.25035  -2.853 0.004329 **
## chol        -0.15432    0.26575  -0.581 0.561440
## fbs          0.51586    0.66070   0.781 0.434930
## restecg      0.12280    0.48338   0.254 0.799466
## thalach      0.87541    0.32238   2.715 0.006618 **
## exang       -1.14520    0.51804  -2.211 0.027062 *
## oldpeak     -0.30130    0.36308  -0.830 0.406621
## slope        0.85293    0.47367   1.801 0.071752 .
## ca          -1.16882    0.30204  -3.870 0.000109 ***
## thal         1.57769    0.51184   3.082 0.002054 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 277.26  on 199  degrees of freedom
## Residual deviance: 125.41  on 186  degrees of freedom
## AIC: 153.41
##
## Number of Fisher Scoring iterations: 6
```

**7) Use the model in 6 to predict the test data and use cutoffs of .5, .6 and .7.** First, use the 0.5 cutoff:

```
testdata = heartdata[c(101:203),]
outtest = predict(model1red, newdata = testdata, type = "response")
library(e1071)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
confusionMatrix(data = as.factor(as.numeric(outtest>0.5)), reference = as.factor(testdata$targe
t))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28 10
##          1 10 55
##
##                Accuracy : 0.8058
##                  95% CI : (0.7162, 0.8772)
##     No Information Rate : 0.6311
##     P-Value [Acc > NIR] : 9.442e-05
##
##                   Kappa : 0.583
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.7368
##             Specificity : 0.8462
##          Pos Pred Value : 0.7368
##          Neg Pred Value : 0.8462
##              Prevalence : 0.3689
##          Detection Rate : 0.2718
##    Detection Prevalence : 0.3689
##       Balanced Accuracy : 0.7915
##
##        'Positive' Class : 0
##
```

Now use the 0.6 cutoff:

```
confusionMatrix(data = as.factor(as.numeric(outtest>0.6)), reference = as.factor(testdata$targe
t))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 31 14
##          1  7 51
##
##                Accuracy : 0.7961
##                  95% CI : (0.7054, 0.8691)
##     No Information Rate : 0.6311
##     P-Value [Acc > NIR] : 0.000227
##
##                   Kappa : 0.5783
##
##  Mcnemar's Test P-Value : 0.190430
##
##             Sensitivity : 0.8158
##             Specificity : 0.7846
##          Pos Pred Value : 0.6889
##          Neg Pred Value : 0.8793
##              Prevalence : 0.3689
##          Detection Rate : 0.3010
##    Detection Prevalence : 0.4369
##       Balanced Accuracy : 0.8002
##
##        'Positive' Class : 0
##
```

Now use the 0.7 cutoff:

```
confusionMatrix(data = as.factor(as.numeric(outtest>0.7)), reference = as.factor(testdata$targe
t))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 31 15
##          1  7 50
##
##                Accuracy : 0.7864
##                  95% CI : (0.6947, 0.861)
##     No Information Rate : 0.6311
##     P-Value [Acc > NIR] : 0.0005159
##
##                   Kappa : 0.5605
##
##  Mcnemar's Test P-Value : 0.1355930
##
##             Sensitivity : 0.8158
##             Specificity : 0.7692
##          Pos Pred Value : 0.6739
##          Neg Pred Value : 0.8772
##              Prevalence : 0.3689
##          Detection Rate : 0.3010
##    Detection Prevalence : 0.4466
##       Balanced Accuracy : 0.7925
##
##        'Positive' Class : 0
##
```

The results are fairly similar. Using a cutoff value of 0.6 has the highest combination of sensitivity and specificity at 1.60, which is one measure of goodness of prediction shown in our lecture slides. Using a cutoff value of 0.6 or 0.7 both get a higher sensitivity of 0.8158. The cutoff of 0.5 results in a higher specificity value of 0.8462. But we want both measures to be high (sensitivity = proportion of observed positives that were predicted positive, specificity = proportion of observed negatives predicted to be negative).

**8) Draw ROC curve for test data in question 7.**

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
roccurve = roc(testdata$target ~ outtest)
```

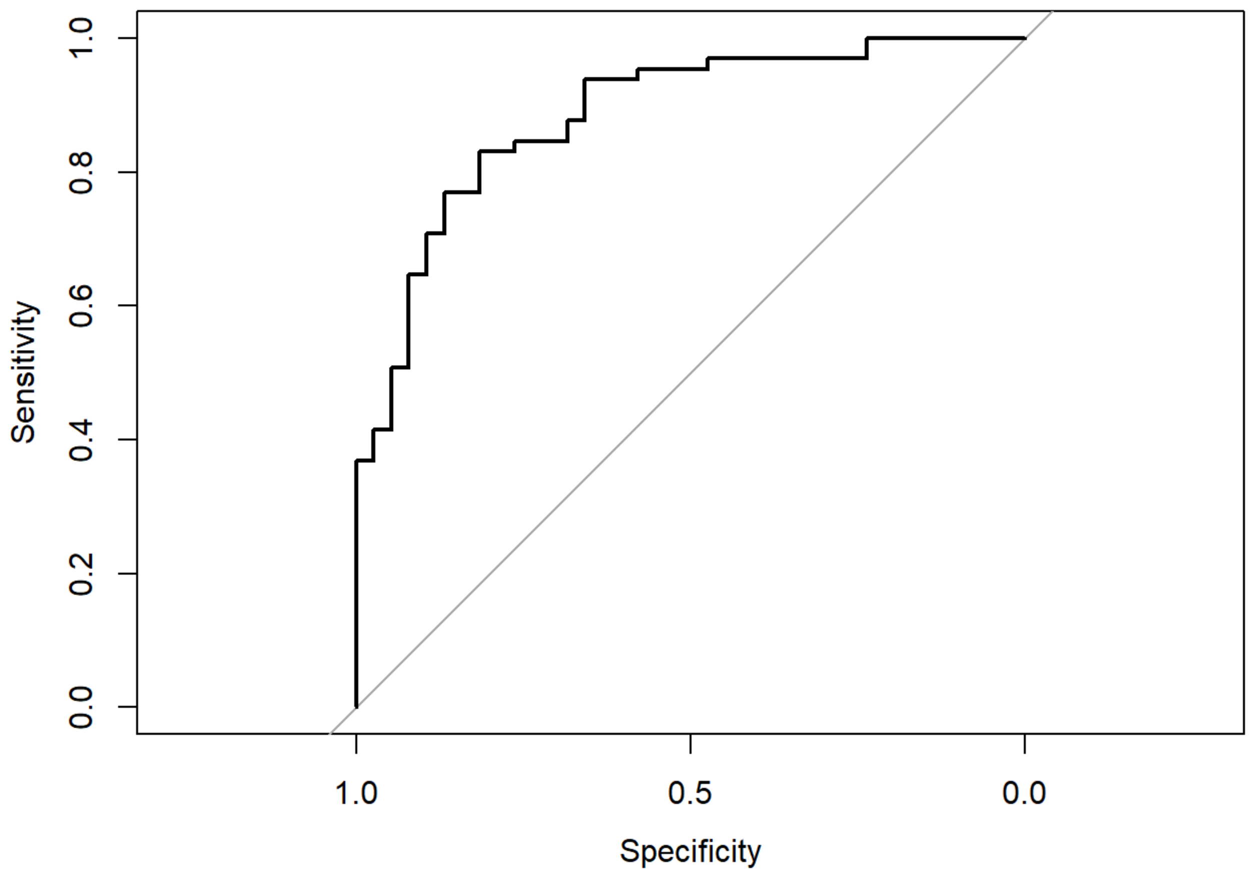
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
roccurve
```

```
##
## Call:
## roc.formula(formula = testdata$target ~ outtest)
##
## Data: outtest in 38 controls (testdata$target 0) < 65 cases (testdata$target 1).
## Area under the curve: 0.8846
```

```
plot(roccurve)
```



The AUC (area under the curve) is fairly high at 0.885. Values above 0.7 indicate a model has moderate to good discriminatory power, so this model has good discriminatory power.

**9) Redo questions 6 and 8 but remove ca, cp and thal as explanatory variables. Evaluate discriminatory power of this new model.**

```
model9red = glm(target~.-ca-cp-thal, data = smallerdata, family="binomial")
summary(model9red)
```

```
##
## Call:
## glm(formula = target ~ . - ca - cp - thal, family = "binomial",
##     data = smallerdata)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.45728  -0.68763   0.02756   0.69531   2.41407
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.20929    0.69102   1.750 0.080118 .
## age         -0.24863    0.22390  -1.110 0.266809
## sex         -1.64947    0.48201  -3.422 0.000622 ***
## trestbps    -0.61118    0.21471  -2.847 0.004420 **
## chol        -0.27081    0.19502  -1.389 0.164953
## fbs          0.49964    0.50171   0.996 0.319312
## restecg     -0.01974    0.38507  -0.051 0.959113
## thalach      0.98919    0.26359   3.753 0.000175 ***
## exang       -1.51737    0.40648  -3.733 0.000189 ***
## oldpeak     -0.39428    0.24781  -1.591 0.111594
## slope        0.31688    0.38424   0.825 0.409536
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 277.26  on 199  degrees of freedom
## Residual deviance: 176.15  on 189  degrees of freedom
## AIC: 198.15
##
## Number of Fisher Scoring iterations: 5
```

```
outtest9 = predict(model9red, newdata = testdata, type = "response")
roccurve9 = roc(testdata$target ~ outtest9)
```

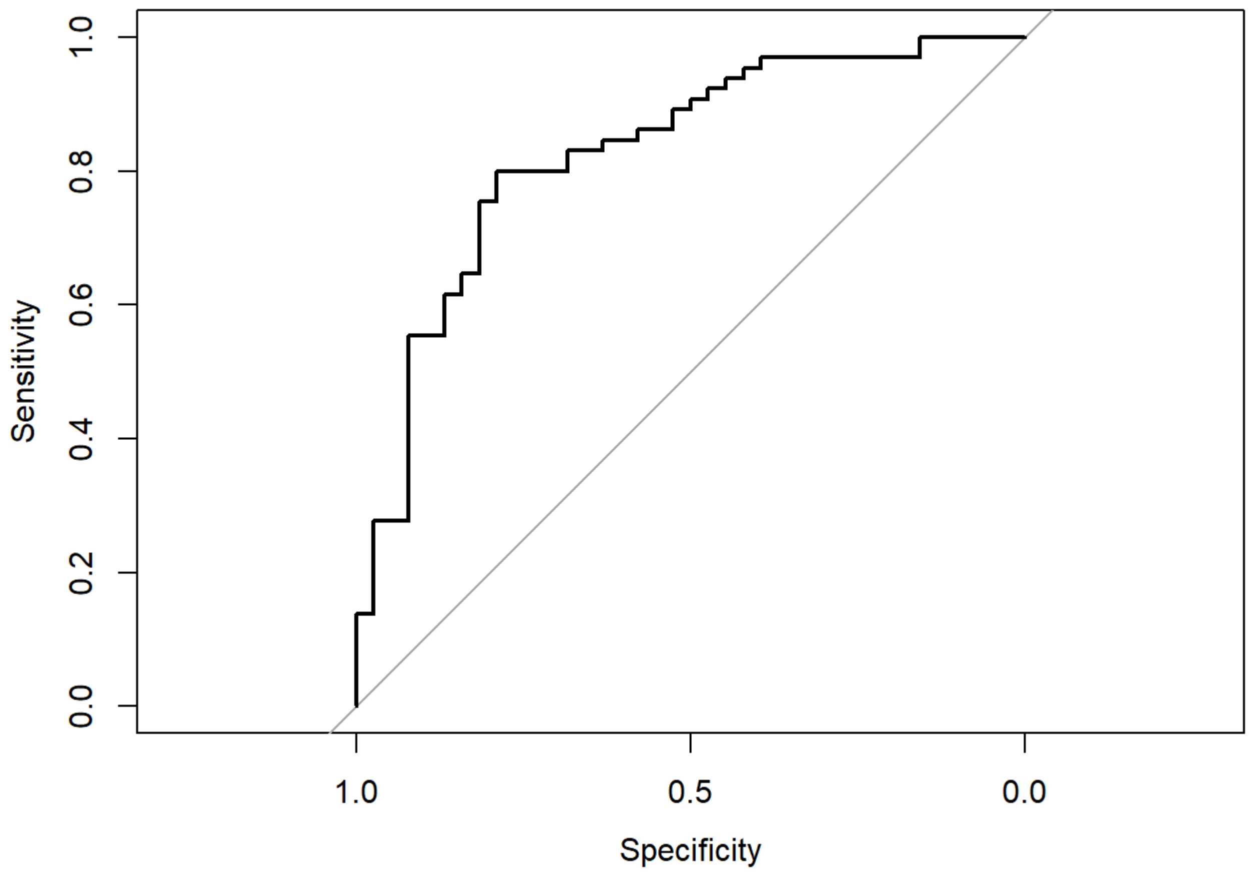
```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
roccurve9
```

```
##
## Call:
## roc.formula(formula = testdata$target ~ outtest9)
##
## Data: outtest9 in 38 controls (testdata$target 0) < 65 cases (testdata$target 1).
## Area under the curve: 0.8271
```

```
plot(roccurve9)
```



This reduced model has a reduced area under the ROC curve of 0.827 (as expected given fewer explanatory variables). However, this is still well above our cutoff value of 0.7 which indicates a model with moderate to good discriminatory power.