PRACTICAL 7: Practical of Logistic Regression

step 1: data file – binary.csv

- This dataset has a binary response (outcome, dependent) variable called **admit**. There are three predictor variables: gre, gpa and rank. We will treat the variables gre and gpa as continuous. The variable rank takes on the values 1 through 4. Institutions with a rank of 1 have the highest prestige, while those with a rank of 4 have the lowest. We can get basic descriptives for the entire data set by using summary.

```
> data <- read.csv(file.choose(), header=T, sep=",")
> head(data)
  admit gre  gpa rank
1     0 380 3.61    3
2     1 660 3.67    3
3     1 800 4.00    1
4     1 640 3.19    4
5     0 520 2.93    4
6     1 760 3.00    2
> summary(data)
     admit             gre             gpa             rank
 Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000
 1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000
 Median :0.0000   Median :580.0   Median :3.395   Median :2.000
 Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485
 3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000
 Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000
> str(data)
'data.frame':   400 obs. of  4 variables:
 $ 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 ...
```

step 2: logistic regression – creating the model

- The code below estimates a logistic regression model using the glm (generalized linear model) function. First, we convert rank to a factor to indicate that rank should be treated as a categorical variable.

```
> data$rank <- as.factor(data$rank)
> str(data)
'data.frame':   400 obs. of  4 variables:
 $ 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 : Factor w/ 4 levels "1","2","3","4": 3 3 1 4 4 2 1 2 3 2 ...
> names(data)
[1] "admit" "gre"   "gpa"   "rank"
> model1 <- glm(admit ~ gre + gpa + rank, data = data, family = "binomial")
```

```
> summary(model1)

Call:
glm(formula = admit ~ gre + gpa + rank, family = "binomial",
    data = data)

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 *
rank2       -0.675443   0.316490  -2.134 0.032829 *
rank3       -1.340204   0.345306  -3.881 0.000104 ***
rank4       -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
```

- In the output above, the first thing we see is the call, this is R reminding us what the model we ran was, what options we specified, etc.
- Next we see the deviance residuals, which are a measure of model fit. This part of output shows the distribution of the deviance residuals for individual cases used in the model. Below we discuss how to use summaries of the deviance statistic to assess model fit.
- The next part of the output shows the coefficients, their standard errors, the z-statistic (sometimes called a Wald z-statistic), and the associated p-values. Both gre and gpa are statistically significant, as are the three terms for rank. The logistic regression coefficients give the change in the log odds of the outcome for a one unit increase in the predictor variable.
    - For every one unit change in gre, the log odds of admission (versus non-admission) increases by 0.002.
    - For a one unit increase in gpa, the log odds of being admitted to graduate school increases by 0.804.
    - The indicator variables for rank have a slightly different interpretation. For example, having attended an undergraduate institution with rank of 2, versus an institution with a rank of 1, changes the log odds of admission by -0.675.
- Below the table of coefficients are fit indices, including the null and deviance residuals and the AIC.

step 3: global testing for the acceptance of the model

```
> null <- glm(admit~1, family = binomial, data=data)
> anova(null, model1, test="Chisq")
Analysis of Deviance Table

Model 1: admit ~ 1
Model 2: admit ~ gre + gpa + rank
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1       399     499.98
2       394     458.52  5   41.459 7.578e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

step 4: predicting the probabilities

```
> data$predprob <- round(fitted(model1),2)
> head(data)
  admit gre  gpa rank predprob
1     0 380 3.61    3     0.17
2     1 660 3.67    3     0.29
3     1 800 4.00    1     0.74
4     1 640 3.19    4     0.18
5     0 520 2.93    4     0.12
6     1 760 3.00    2     0.37
```

- We predict probabilities using fitted method and round the probability value to 2.
- looking at the predprob column in the output:
  - o the predicted probability of being accepted into a graduate program is 0.74 (highest) for students from the highest prestige undergraduate institutions (rank=1
  - o and 0.12 for students from the lowest ranked institutions (rank=4), holding gre and gpa at their means.

step 5: classification and misclassification analysis

install.packages("gmodels")

```
> library(gmodels)
> tab <- table(data$admit, fitted(model1)>0.5)
> tab

    FALSE TRUE
  0   254   19
  1    97   30
```

conclusion: we can conclude following things from the table (confusion matrix):

1. 254 students were not admitted and the model also predicts that they should not be admitted. This is correct classification.
2. 97 students were not actually admitted but model predicts them to be admitted. This is misclassification.
3. 19 students were admitted but the model predicts that they should not be admitted. This is again misclassification.
4. 30 students were admitted and model also predicts them to be admitted. This is correct classification.

```
> sum(diag(tab))/sum(tab)
[1] 0.71
> 1-sum(diag(tab))/sum(tab)
[1] 0.29
```

The correct classification is 71% while there is 29% of misclassification by the model.

```
#check the trade-off between sensitivity and specificity using different cut values
> table(data$admit, fitted(model1)>0.1)

    FALSE TRUE
  0     9  264
  1     0  127
> table(data$admit, fitted(model1)>0.2)

    FALSE TRUE
  0    83  190
  1    18  109
> table(data$admit, fitted(model1)>0.3)

    FALSE TRUE
  0   161  112
  1    42   85
> table(data$admit, fitted(model1)>0.4)

    FALSE TRUE
  0   224   49
  1    71   56
> table(data$admit, fitted(model1)>0.5)

    FALSE TRUE
  0   254   19
  1    97   30
```

step 6: model performance evaluation

```
#goodness of fit using receiver Operational Curve
#use plot to check proper cutoff point
#use exp(coef(model1)) to check coefficients
```

- The prediction and performance functions are the workhorses of most of the analyses in ROCR where predictions are some predicted measure (usually continuous) for the "truth".
- In the performance object, we see that the first argument is a prediction object, and the second is a measure (here it is tpr- true positive rate and fpr- false positive rate).
- We will do an ROC curve (**Receiver Operating Characteristic curve)**, which plots the false positive rate (FPR) on the x-axis and the true positive rate (TPR) on the y-axis.
- An ROC curve demonstrates several things:
  - o It shows the tradeoff between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity). The closer the curve follows the

left-hand border and then the top border of the ROC space, the more accurate the test.
- The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.

```
> pred <- predict(model1,data,type="response")
> library(ROCR)
Loading required package: gplots

Attaching package: ‘gplots’

The following object is masked from ‘package:stats’:

    lowess

Warning messages:
1: package ‘ROCR’ was built under R version 3.5.3
2: package ‘gplots’ was built under R version 3.5.3
> data$predprob<-fitted(model1)
> rocrpred<-prediction(pred, data$admit)
> rocrperf<-performance(rocrpred,"tpr","fpr")
> plot(rocrperf,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))
```

```
> coef(model1)
 (Intercept)          gre          gpa        rank2        rank3        rank4 
-3.989979073  0.002264426  0.804037549 -0.675442928 -1.340203916 -1.551463677 
> exp(coef(model1))
(Intercept)         gre         gpa       rank2       rank3       rank4 
  0.0185001   1.0022670   2.2345448   0.5089310   0.2617923   0.2119375 
```

Output of plot: