Logistic regression

Self-test answers

- We learnt how to do hierarchical regression in the previous chapter. Try to conduct a hierarchical logistic regression analysis on these data. Enter **Previous** and **PSWQ** in the first block and **Anxious** in the second.

First of all, as always, load in the data (once you have set your working directory to be the place where you have saved the **penalty.dat** file of course ☺):

```
penaltyData<-read.delim("penalty.dat", header = TRUE)
```

We can see the first six cases of data by executing:

```
head(penaltyData)
```

```
PSWQ Anxious Previous         Scored
1   18      21       56 Scored Penalty
2   17      32       35 Scored Penalty
3   16      34       35 Scored Penalty
4   14      40       15 Scored Penalty
5    5      24       47 Scored Penalty
6    1      15       67 Scored Penalty
```

Looking at the data above, we can see that the categorical variable **Scored** has two categories, *Missed penalty* and *Scored penalty* and **R** has encoded it as a factor. If you think back to the eel example in the chapter, you will remember that **R** encodes factors in alphabetical order, and this doesn't always make the most sense for our data analysis. However, in this particular example it is not a problem because **Missed** will be category 1 and **Scored** category 2. In other words, **Missed** will be the baseline category, which is what we want. As such, we don't need to set the baseline category like I did in the book chapter for this example, hooray!

The next thing that we need to do is to create the two hierarchical models. We can do this by using the *glm()* function as we did in the chapter. We are carrying out a hierarchical regression: in model 1 we want to include both **Previous** and **PSWQ** because these are previously established predictors and so it is a good idea to enter them into the model in a single block, and then in model 2 we want to add **Anxious**. To create the first model we can execute:

```
penaltyModel.1 <- glm(Scored ~ Previous + PSWQ, data = penaltyData, family =
binomial())
```

This command creates a model called *penaltyModel.1* in which **Scored** is predicted from **Previous** and **PSWQ**. Similarly, we can create the second model by executing:

```
penaltyModel.2 <- glm(Scored ~ Previous + PSWQ + Anxious, data = penaltyData, family =
binomial())
```

This command creates a model called *penaltyModel.2* in which **Scored** is predicted from **Previous**, **PSWQ** and **Anxious**. We could also have created *penaltyModel.2* by executing the shorter command:

```
penaltyModel.2 <- update(penaltyModel.1, .~. + Anxious)
```

To see the models that we have just generated we need to execute the *summary()* function (remembering to put the model name into the function):

```
summary(penaltyModel.1)
```

```
summary(penaltyModel.2)
```

```
> summary(penaltyModel.1)

Call:
glm(formula = Scored ~ Previous + PSWQ, family = binomial(),
    data = penaltyData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2212  -0.3306   0.1038   0.5046   1.6067

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.28031    1.67017   0.767  0.44333
Previous     0.06480    0.02209   2.934  0.00335 **
PSWQ        -0.23009    0.07983  -2.882  0.00395 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 103.638  on 74  degrees of freedom
Residual deviance:  48.662  on 72  degrees of freedom
AIC: 54.662

Number of Fisher Scoring iterations: 6
```

The output above shows the model summary for the model 1 (*penaltyModel.1*). We will begin by looking at the summary statistics about the model. The overall fit of the model is assessed using the deviance statistic. Remember that larger values of the deviance statistic indicate poorly fitting statistical models. **R** provides two deviance statistics: the null deviance, and the residual deviance. The null deviance is the deviance of the model that contains no predictors, other than the constant. The residual deviance is the deviance for the model.

At this stage of the analysis the value of the deviance for the model should be less than the value for the null model (when only the constant was included in the model) because lower values of $-2LL$ indicate that the model is predicting the outcome variable more accurately. For the null model , $-2LL$ = 103.64, but when **Previous** and **PSWQ** have been included this value has been reduced to 48.66. This reduction tells us that the model is better at predicting whether someone will score a penalty than it was before **Previous** and **PSWQ** were added.

The question of how much better the model predicts the outcome variable can be assessed using the *model chi-square statistic*, which measures the difference between the model as it currently stands and the model when only the constant was included. We can use **R** to automatically calculate the model chi-square and its significance for us by executing:

```
modelChi <- penaltyModel.1$null.deviance - penaltyModel.1$deviance

chidf <- penaltyModel.1$df.null - penaltyModel.1$df.residual

chisq.prob <- 1 - pchisq(modelChi, chidf)
```

We can then view the output of these commands by executing:

```
modelChi; chidf; chisq.prob
```

```
[1] 54.97669
[1] 2
[1] 1.1533e-12
```

The change in the amount of information explained by the model (54.98) is significant (p < .0001), and so using previous experience and worry as predictors significantly improves our ability to predict penalty success, $\chi^2(2) = 54.98$, $p < .0001$.

Next, we consider the coefficients. This part is crucial because it tells us the estimates for the coefficients for the predictors included in the model. This section of the output gives us the coefficients and statistics for the variables that have been included in the model at this point (namely **Previous**, **PSWQ** and the constant). The crucial statistic is the z-statistic, which has a normal distribution and tells us whether the b coefficient for that predictor is significantly different from zero. If the coefficient is significantly different from zero then we

can assume that the predictor is making a significant contribution to the prediction of the outcome. For these data it seems to indicate that previous experience, $b = 0.06$, $z = 2.93$, $p < .01$, and worry, $b = -0.23$, $z = -2.88$, $p < .01$, are both significant predictors of penalty success (note that the significance of the z-statistics are both less than .05).

Next, we want to calculate the various values of R^2, which we can do by executing (remember that you need to have executed the function code from the book chapter first):

```
logisticPseudoR2s(penaltyModel.1)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.53
Cox and Snell R^2         0.52
Nagelkerke R^2            0.694
```

As you can see, all of the values of R^2 differ slightly, but they can be used as effect size measures for the model. We can interpret the result of Hosmer and Lemeshow's goodness-of-fit test as meaning that the model can account for 53% of the variance in penalty success (so, roughly half of what makes a penalty kick successful is still unknown).

We can next calculate the odds ratio as the exponential of the *b* coefficient for the predictor variables. These coefficients are stored in a variable called *coefficients*, which is part of the model we created. Therefore, we can access this variable as:

```
penaltyModel.1$coefficients
```

This just means 'the variable called *coefficients* within the model called *eelModel.1*'. It's a simple matter to apply the *exp()* function to this variable to find out the odds ratio:

```
exp(penaltyModel.1$coefficients)
```

Executing this command will display the odds ratio for the predictors in the model:

```
(Intercept)    Previous        PSWQ
  3.5977600   1.0669482   0.7944584
```

The value of the odds ratio for **previous** indicates that if the percentage of previous penalties scored goes up by one, then the odds of scoring a penalty also increase (because the odds ratio is greater than 1). The odds ratio for **PSWQ** indicates that if the level of worry increases by one point along the Penn State worry scale, then the odds of scoring a penalty decrease (because it is less than 1).

We can also calculate confidence intervals for the odds ratios. To obtain confidence intervals of the parameters, we use the *confint()* function – just as we did for ordinary regression. We can also exponentiate these with the *exp()* function. To get the confidence intervals execute:

```
exp(confint(penaltyModel.1))
```

```
2.5 %      97.5 %
(Intercept) 0.1379603 108.7408589
Previous    1.0263545   1.1216277
PSWQ        0.6614590   0.9115488
```

The confidence interval for **Previous** ranges from 1.02 to 1.12, so we can be very confident that the value of the odds ratio in the population lies somewhere between these two values. What's more, because both values are greater than 1 we can also be confident that the relationship between **Previous** and penalty success found in this sample is true of the whole population of footballers. The confidence interval for **PSWQ** ranges from .66 to .91, so we can be very confident that the value of the odds ratio in the population lies somewhere between these two values. In addition, because both values are less than 1, we can be confident that the relationship between **PSWQ** and penalty success found in this sample is true of the whole population of footballers. If we had found that the confidence interval ranged from less than 1 to more than 1, then this would limit the generalizability of our findings because the odds ratio in the population could indicate either a positive (odds ratio > 1) or negative (odds ratio < 1) relationship.

The output for model 2 (below) shows what happens to the model when our new predictor (**Anxious**) is added. The effect of adding **Anxious** to the model is to reduce the –2*LL* to 47.416 (a reduction of 1.246 from model 1, suggesting that including **Anxious** in the model has not improved our ability to predict whether a penalty will be scored or missed. In addition, we can see that the AIC is higher in model 2 (55.416) than model 1 (54.66), indicating that model 1 is the better model.

```
Call:
glm(formula = Scored ~ Previous + PSWQ + Anxious, family = binomial(),
    data = penaltyData)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.31374  -0.35996   0.08334   0.53860   1.61380

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -11.49256   11.80175  -0.974  0.33016
Previous      0.20261    0.12932   1.567  0.11719
PSWQ         -0.25137    0.08401  -2.992  0.00277 **
Anxious       0.27585    0.25259   1.092  0.27480
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 103.638  on 74  degrees of freedom
Residual deviance:  47.416  on 71  degrees of freedom
AIC: 55.416
```

We can again compare the models by finding the difference in the deviance statistics. This difference is chi-square distributed. We can find this difference in two ways. First, we can subtract one deviance from the other as we did before. An easier method, though, is to use the *anova()* function. The *anova()* function has the advantage that it also calculates the degrees of freedom for us. If we do the calculations manually we can use the same commands as before, except that rather than using the *null.deviance* and *df.null* variables, we use the *deviance* and *df.residual* variables for the two models we're comparing – in each case we subtract model 2 from model 1:

```
modelChi <- penaltyModel.1$deviance - penaltyModel.2$deviance

chidf <- penaltyModel.1$df.residual - penaltyModel.2$df.residual

chisq.prob <- 1 - pchisq(modelChi, chidf)

modelChi; chidf; chisq.prob
```

```
[1] 1.246267
[1] 1
[1] 0.2642667
```

You should find that the difference between the models (*modelChi*) is 1. 246267, with one degree of freedom (*chidf*), and a *p*-value (*chisq.prob*) of .2642667. As this value is greater than .05, we can conclude that model 2 (with **Anxious** added as a predictor) is not a significant improvement over model 1 (which included only **Previous** and **PSWQ** as predictors).

We can do the same with the *anova()* function. Remember that with this function we simply list the models in the order in which we want to compare them. Therefore, to compare our two models we would execute:

```
anova(penaltyModel.1, penaltyModel.2)
```

```
Analysis of Deviance Table

Model 1: Scored ~ Previous + PSWQ
Model 2: Scored ~ Previous + PSWQ + Anxious
```

```
  Resid. Df   Resid. Dev  Df  Deviance
1        72     48.662
2        71     47.416    1    1.2463
```

The coefficients part of the output now contains all three predictors and something very interesting has happened: **PSWQ** is still a significant predictor of penalty success; however, **Previous** experience no longer significantly predicts penalty success. In addition, state anxiety appears not to make a significant contribution to the prediction of penalty success. How can it be that previous experience no longer predicts penalty success, and neither does anxiety, yet the ability of the model to predict penalty success has improved slightly?

Let's again compute R^2 for model 2 (remember to execute the function code first if you haven't done so already):

```
logisticPseudoR2s(penaltyModel.2)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.542
Cox and Snell R^2         0.527
Nagelkerke R^2            0.704
```

We can next calculate the odds ratio by executing:

```
exp(penaltyModel.2$coefficients)
```

Executing this command will display the odds ratio for the predictors in the model:

```
(Intercept)     Previous         PSWQ      Anxious
1.020573e-05 1.224593e+00 7.777351e-01 1.317649e+00
```

We can also calculate confidence intervals for the odds ratios, as we did for model 1:

```
exp(confint(penaltyModel.2))
```

```
                   2.5 %       97.5 %
(Intercept) 1.876236e-16 6.425863e+04
Previous    9.607908e-01 1.612961e+00
PSWQ        6.420582e-01 8.988251e-01
Anxious     8.138225e-01 2.242252e+00
```

If we examine the values of the odds ratio for both **Previous** and **Anxious** it is clear that they both potentially have a positive relationship to penalty success (i.e. as they increase by a unit, the odds of scoring improve). However, the confidence intervals for these values cross 1, which indicates that the direction of this relationship may be unstable in the population as a whole (i.e. the value of the odds ratio in our sample may be quite different to the value if we had data from the entire population).

You may be tempted to use this final model to say that, although worry is a significant predictor of penalty success, the previous finding that experience plays a role is incorrect. This would be a dangerous conclusion to make, and if you read the section on multicollinearity in the book you'll see why.

- Using what you learned in Chapter 6, carry out a Pearson correlation between all of the variables in this analysis. Can you work out why we have a problem with collinearity?

You can execute this command:

```
cor(penaltyData[, c("Previous", "PSWQ", "Anxious")])
```

This applies the *cor()* function to the predictor variables in the *penaltyData* dataframe. The results of your analysis should look like this:

```
           Previous       PSWQ    Anxious
Previous  1.0000000 -0.6435448 -0.9928699
PSWQ     -0.6435448  1.0000000  0.6516416
Anxious  -0.9928699  0.6516416  1.0000000
```

From this output we can see that **Anxious** and **Previous** are highly negatively correlated (*r* = −.99); in fact they are nearly perfectly correlated. Both **Previous** and **Anxious** correlate with penalty success but because they are correlated so highly with each other, it is unclear which of the two variables predicts penalty success in the regression. As such our multicollinearity stems from the near-perfect correlation between **Anxious** and **Previous**.

- Try creating two new variables that are the natural log of **Anxious** and **Previous**. (Remember that 0 has no log, so if any of the variables have a zero, you'll need to add a constant – see section 5.8.3.2.)

```
penaltyData$logAnxInt<-log(penaltyData$Anxious)*penaltyData$Anxious
```

This command creates a new variable called *logAnxInt* in the *penaltyData* dataframe that is the variable **Anxious** (*penaltyData$Anxious*) multiplied by the log of that variable (*log(penaltyData$Anxious)*).

```
penaltyData$logPrevInt<-log(penaltyData$Previous + 1)*penaltyData$Previous
```

This command creates a new variable called *logPrevInt* in the *penaltyData* dataframe that is the variable **Previous** (*penaltyData$Previous*) multiplied by the log of that variable plus 1 (*log(penaltyData$Previous + 1)*). We included the '+1' because this variable contained 0 values (and there is no log of zero).

- What does the log-likelihood measure?

The log-likelihood statistic is analogous to the residual sum of squares in multiple regression in the sense that it is an indicator of how much unexplained information there is after the model has been fitted. It follows, therefore, that large values of the log-likelihood statistic indicate poorly fitting statistical models, because the larger the value of the log-likelihood, the more unexplained observations there are.

- Use what you learnt earlier in this chapter to check the assumptions of multicollinearity and linearity of the logit.

Testing for linearity of the logit

In this example we have three continuous variables (**Funny**, **Sex**, **Good_Mate**), therefore we have to check that each one is linearly related to the log of the outcome variable (**Success**). To test this assumption we need to run the logistic regression but include predictors that are the interaction between each predictor on the log of itself. For each variable, create a new variable that is the log of the original variable. We need to create the interaction terms of each of the variables with its log, by creating using the *log()* function. We create these variables by executing:

```
mlChat$logFunny<-log(mlChat$Funny +1)
mlChat$logGood<-log(mlChat$Good_Mate +1)
mlChat$logSex<-log(mlChat$Sex + 1)
```

These commands create three new variables in the *mlChat* dataframe that reflect the log of each predictor.

To test the assumption we need to redo the analysis exactly the same as before, except that we should put all variables in a single block (i.e., we don't need to do it hierarchically), and we also need to put in three new interaction terms, consisting of each predictor and their logs. We create the model by executing:

```
chatTest.1 <- mlogit(Success ~ 1 | Good_Mate + Funny + Sex + Funny:logFunny +
Good_Mate:logGood + Sex:logSex, data = mlChat, reflevel=3)
summary(chatTest.1)
```

This command creates a model (*chatTest.1*) in which the variable **Success** is predicted from **Good_Mate**, **Funny**, **Sex**, the interaction between **Good_Mate** and its log (*Good_Mate:logGood),* the interaction between **Funny** and its log (*Funny:logFunny)* and the interaction between **Sex** and its log (*Sex:logSex).* We then use the *summary()* function to display the model.

```
Coefficients :
                                          Estimate Std. Error t-value  Pr(>|t|)
altGet Phone Number                      -0.107256   1.104480 -0.0971 0.9226394
altGo Home with Person                    2.311981   1.409535  1.6402 0.1009544
altGet Phone Number:Good_Mate            -0.204173   0.406084 -0.5028 0.6151149
altGo Home with Person:Good_Mate         -0.881675   0.522739 -1.6866 0.0916719 .
altGet Phone Number:Funny                -0.585111   0.487429 -1.2004 0.2299831
altGo Home with Person:Funny             -1.785351   0.607114 -2.9407 0.0032745 **
altGet Phone Number:Sex                   0.103035   0.419337  0.2457 0.8059067
altGo Home with Person:Sex               -1.485951   0.491596 -3.0227 0.0025053 **
altGet Phone Number:Funny:logFunny        0.300997   0.201528  1.4936 0.1352873
altGo Home with Person:Funny:logFunny     0.850784   0.244860  3.4746 0.0005117 ***
altGet Phone Number:Good_Mate:logGood     0.133730   0.156107  0.8567 0.3916369
altGo Home with Person:Good_Mate:logGood  0.395804   0.201207  1.9671 0.0491663 *
altGet Phone Number:Sex:logSex           -0.026916   0.161473 -0.1667 0.8676130
altGo Home with Person:Sex:logSex         0.623765   0.189443  3.2926 0.0009926 ***
```

The output above is all that we need to look at because it tells us about whether any of our predictors significantly predict the outcome categories. The assumption of linearity of the logit is tested by the six interaction terms, the majority of which are significant ($p < .05$), which means that *all three predictors have violated the assumption*.

Testing for multicollinearity

First, if you haven't already, read the data into a new dataframe, which we'll call *chatData*, by setting your working directory to the location of the file and executing:

```
chatData<-read.delim("Chat-Up Lines.dat", header = TRUE)
```

Set *Female* to be the baseline category of **Gender**:

```
chatData$Gender<-relevel(chatData$Gender, ref = 2)
```

Now we need to create a model using the *chatData* in its original format, with all four predictors:

```
chatModel <- glm(Success ~ Funny + Good_Mate + Sex + Gender, data = chatData, family =
binomial())
```

Having created this model, we can get the VIF and tolerance as we did in Chapter 7 by entering the model name into the *vif()* function from the *car* package. Execute:

```
vif(chatModel)
```

```
1/vif(chatModel)
```

The first line gives you the VIF values and the second the tolerance (which is simply the reciprocal of the VIF).

```
Funny     Good_Mate  Sex       Gender
1.304700  1.027076   1.017565  1.262266

Funny     Good_Mate  Sex       Gender
0.7664597 0.9736373  0.9827379 0.7922259
```

The results are shown in the output above (first VIF and then tolerance). Menard (1995) suggests that a tolerance value less than 0.1 almost certainly indicates a serious collinearity problem. Myers (1990) also suggests that a VIF value greater than 10 is cause for concern and in these data all of the VIFs are well below 10 (and tolerances above 0.1). It seems from these values that there is not an issue of collinearity between the predictor variables.

We can investigate this issue further by examining the correlations between the numeric variables by executing:

```
cor(chatData[,c("Funny", "Good_Mate", "Sex")])
```

```
            Funny    Good_Mate   Sex
Funny      1.0000000 0.16320983 0.11560845
Good_Mate 0.1632098 1.00000000 0.03794612
Sex       0.1156084 0.03794612 1.00000000
```

Looking at the correlations above, we can be confident in concluding that there is no problem with multicollinearity in these data as none of the variables correlate very highly, all *r*s < .2.

Oliver Twisted

Please Sir, can I have some more … diagnostics?

```
      leverage studentized.residuals dfbeta.(Intercept) dfbeta.InterventionIntervention
1   0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
2   0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
3   0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
4   0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
5   0.01754386             0.8160435      -3.039582e-17                    3.225994e-02
6   0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
7   0.01754386            -1.6083171      -5.526513e-17                   -6.334765e-02
8   0.01754386             0.8160435      -2.486931e-17                    3.225994e-02
9   0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
10  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
11  0.01754386             0.8160435      -2.486931e-17                    3.225994e-02
12  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
13  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
14  0.01754386            -1.6083171      -4.973862e-17                   -6.334765e-02
15  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
16  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
17  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
18  0.01754386            -1.6083171      -4.973862e-17                   -6.334765e-02
19  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
20  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
21  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
22  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
23  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
24  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
25  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
26  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
27  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
28  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
29  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
30  0.01754386             0.8160435      -2.210605e-17                    3.225994e-02
31  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
32  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
33  0.01754386             0.8160435      -1.934280e-17                    3.225994e-02
34  0.01754386            -1.6083171      -3.868559e-17                   -6.334765e-02
35  0.01754386             0.8160435      -1.934280e-17                    3.225994e-02
36  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
37  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
38  0.01754386            -1.6083171      -3.315908e-17                   -6.334765e-02
39  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
40  0.01754386             0.8160435      -1.934280e-17                    3.225994e-02
41  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
42  0.01754386            -1.6083171      -3.315908e-17                   -6.334765e-02
43  0.01754386             0.8160435      -1.934280e-17                    3.225994e-02
44  0.01754386            -1.6083171      -3.315908e-17                   -6.334765e-02
45  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
46  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
47  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
48  0.01754386            -1.6083171      -3.315908e-17                   -6.334765e-02
49  0.01754386            -1.6083171      -3.315908e-17                   -6.334765e-02
50  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
51  0.01754386             0.8160435      -1.657954e-17                    3.225994e-02
52  0.01754386             0.8160435      -1.657954e-17                    3.225994e-02
53  0.01754386             0.8160435      -1.657954e-17                    3.225994e-02
54  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
55  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
56  0.01754386             0.8160435      -1.657954e-17                    3.225994e-02
57  0.01754386             0.8160435      -1.657954e-17                    3.225994e-02
58  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
59  0.01754386             0.8160435      -1.657954e-17                    3.225994e-02
60  0.01754386             0.8160435      -1.657954e-17                    3.225994e-02
61  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
62  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
63  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
64  0.01754386            -1.6083171      -2.210605e-17                   -6.334765e-02
65  0.01754386             0.8160435      -1.381628e-17                    3.225994e-02
66  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
67  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
68  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
69  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
70  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
71  0.01754386             0.8160435      -1.381628e-17                    3.225994e-02
72  0.01785714            -1.0643627      -3.886912e-02                    3.886912e-02
73  0.01754386            -1.6083171      -1.105303e-17                   -6.334765e-02
74  0.01754386             0.8160435      -1.381628e-17                    3.225994e-02
75  0.01754386             0.8160435      -1.381628e-17                    3.225994e-02
76  0.01785714             1.3110447       4.782751e-02                   -4.782751e-02
77  0.01754386             0.8160435      -1.381628e-17                    3.225994e-02
```

78	0.01754386	-1.6083171	-1.105303e-17	-6.334765e-02
79	0.01785714	1.3110447	4.782751e-02	-4.782751e-02
80	0.01754386	-1.6083171	-1.105303e-17	-6.334765e-02
81	0.01785714	1.3110447	4.782751e-02	-4.782751e-02
82	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
83	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
84	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
85	0.01754386	-1.6083171	-1.105303e-17	-6.334765e-02
86	0.01754386	0.8160435	-1.381628e-17	3.225994e-02
87	0.01785714	1.3110447	4.782751e-02	-4.782751e-02
88	0.01754386	-1.6083171	-5.526513e-18	-6.334765e-02
89	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
90	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
91	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
92	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
93	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
94	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
95	0.01754386	-1.6083171	0.000000e+00	-6.334765e-02
96	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
97	0.01785714	1.3110447	4.782751e-02	-4.782751e-02
98	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
99	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
100	0.01754386	0.8160435	-1.105303e-17	3.225994e-02
101	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
102	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
103	0.01754386	0.8160435	-8.289770e-18	3.225994e-02
104	0.01785714	1.3110447	4.782751e-02	-4.782751e-02
105	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
106	0.01785714	1.3110447	4.782751e-02	-4.782751e-02
107	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
108	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
109	0.01754386	0.8160435	-2.763257e-18	3.225994e-02
110	0.01754386	0.8160435	-2.763257e-18	3.225994e-02
111	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
112	0.01785714	-1.0643627	-3.886912e-02	3.886912e-02
113	0.01754386	0.8160435	-2.763257e-18	3.225994e-02

Labcoat Leni's real research

Mandatory suicide?

Problem

Lacourse, E. et al. (2001). *Journal of Youth and Adolescence*, *30*, 321–332.

Although I have fairly ecclectic tastes in music, my favourite kind of music is heavy metal. One thing that is mildly irritating about liking heavy is that everyone assumes that you're a miserable or aggressive bastard. When not listening (and often while listening to) heavy metal, I spend most of my time researching clinical psychology: I research how anxiety develops in children. Therefore, I was literally beside myself with excitement when a few years back I stumbled on a paper that combined these two interests. Lacourse, Claes, and Villeneuve (2001) carried out a study to see whether a love of heavy metal could predict suicide risk. Fabulous stuff!

Eric Lacourse and his colleagues used questionnaires to measure several background variables: suicide risk (yes or no), marital status of parents (together or divorced/separated), the extent to which the person's mother and father were neglectful, self-estrangement/powerlessness (adolescents who have negative self-perceptions, are bored with life, etc.), social isolation (feelings of a lack of support), normlessness (beliefs that socially disapproved behaviours can be used to achieve certain goals), meaninglessness (doubting that school is relevant to gaining employment), and drug use. In addition, they measured liking of different categories of music. For heavy metal they included classic bands (Black Sabbath, Iron Maiden), thrash metal bands (Slayer, Metallica), death/black metal bands (Obituary, Burzum) and gothic (Marilyn Manson, Sisters of Mercy). As well as liking, they measured behavioural manifestations of worshipping these bands (hanging posters, hanging out with other metal fans), and vicarious music listening (whether music was used when angry or to bring out aggressive moods). They carried out a logistic regression predicting suicide risk from all of these predictors for males and females separately.

The data for the female sample are in the file **Lacourse et al. (2001) Females.dat**. Labcoat Leni wants you to carry out a logistic regression predicting **Suicide_Risk** from all of the other predictors (forced entry). (To make it easier to compare to the published results I suggest you

enter the predictors in the same order as in Table 3 in the paper: **Age**, **Marital_Status**, **Mother_Negligence**, **Father_Negligence**, **Self_Estrangement**, **Isolation**, **Normlessness**, **Meaninglessness**, **Drug_Use**, **Metal**, **Worshipping**, **Vicarious**.) Create a table of the results; does listening to heavy metal make girls suicidal? If not, what does?

Solution

Let's start by loading in the data, I'm going to call it *suicideData* as it is quicker to type than *LacourseetalFemales.dat*:

```
suicideData<-read.delim("Lacourse et al. (2001) Females.dat", header = TRUE)
```

We have two categorical variables of interest in the *suicideData* dataframe, **Marital_Status** and **Suicide_Risk** (we also have **Gender**, but as all the participants in this dataframe are female we can ignore it). Helpfully, **R** recognizes that these variables are categorical and has loaded them in as factors.

The trouble is that the numbers that **R** has assigned might not be the numbers that we want. In fact, **R** creates levels of the factor by taking the text strings in alphabetical order and assigning them ascending numerical values. In other words, for **Suicide_Risk** we have two categories and **R** will have ordered these categories alphabetically (i.e., 'Non-Suicidal' and 'Suicidal). So, *Non-Suicidal* will be the baseline category because it is first, which is good because this makes sense for our analysis. However, for **Marital_Status** the categories were *Together* and *Separated or Divorced* so given the alphabetic order *Separated or Divorced* will be the baseline category. Yet, it makes more sense to code this variable the opposite way around. It would be good if *Together* was the baseline, or first category, because then we would know that the model coefficients reflect the probability of being separated or divorced (which is what we want to know) rather than the probability of not being divorced.

Fortunately, the function *relevel()* lets us specify the baseline category for a factor. Execute this command:

```
suicideData$Marital_Staus<-relevel(suicideData$Marital_Status, "Together")
```

The variable **Marital_Status** now has *Together* as the first level (i.e. the baseline category). Having set our baseline categories we can get on with the analysis.

The main analysis is fairly simple to specify because we're just putting all predictors in at the same time and so we only need to specify one model (note that I have ordered the predictors as suggested by Labcoat Leni):

```
suicideModel <- glm(Suicide_Risk ~ Age + Marital_Status + Mother_Negligence +
Father_Negligence + Self_Estrangement + Isolation + Normlessness + Meaninglessness +
Drug_Use + Metal + Worshipping + Vicarious, data = suicideData, family = binomial())


summary(suicideModel)
```

```
Call:
glm(formula = Suicide_Risk ~ Age + Marital_Status + Mother_Negligence +
    Father_Negligence + Self_Estrangement + Isolation + Normlessness +
    Meaninglessness + Drug_Use + Metal + Worshipping + Vicarious,
    family = binomial(), data = suicideData)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.78306  -0.56512  -0.29958  -0.07088   2.53406

Coefficients:
                                    Estimate Std. Error z value Pr(>|z|)
(Intercept)                       -19.011598   6.208677  -3.062  0.00220 **
Age                                 0.692853   0.323445   2.142  0.03219 *
Marital_StatusSeparated or Divorced   0.183489   0.677262   0.271  0.78645
Mother_Negligence                  -0.019607   0.053244  -0.368  0.71268
Father_Negligence                   0.084632   0.047858   1.768  0.07699 .
Self_Estrangement                   0.155153   0.064832   2.393  0.01670 *
Isolation                          -0.005822   0.076152  -0.076  0.93906
Normlessness                        0.191307   0.108841   1.758  0.07880 .
Meaninglessness                    -0.066610   0.061024  -1.092  0.27503
```

```
Drug_Use                              0.316855   0.103097   3.073  0.00212 **
Metal                                 0.135552   0.091719   1.478  0.13943
Worshipping                           0.158911   0.129474   1.227  0.21968
Vicarious                            -0.341831   0.196288  -1.741  0.08160 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 135.533  on 120  degrees of freedom
Residual deviance:  85.116  on 108  degrees of freedom
AIC: 111.12

Number of Fisher Scoring iterations: 6
```

The *model chi-square statistic* measures the difference between the model as it currently stands and the model when only the constant was included. We can use **R** to automatically calculate the model chi-square, and its significance to see whether the model is a better fit by executing:

```
modelChi <- suicideModel$null.deviance - suicideModel$deviance
chidf <- suicideModel$df.null - suicideModel$df.residual
chisq.prob <- 1 - pchisq(modelChi, chidf)
```

We can then view the output of the above commands by executing:

```
 modelChi; chidf; chisq.prob
```

```
[1] 50.41736
[1] 12
[1] 1.179907e-06
```

We can report that including all the predictors produced a significant improvement in the fit of the model, $\chi^2(12) = 50.42$, $p < .0001$.

We can calculate the odds ratios as the exponential of the *b* coefficient for the predictor variables using the *exp()* function and by executing:

```
 exp(suicideModel$coefficients)
```

Executing this command will display the odds ratio for the predictors in the model:

```
(Intercept)                 Age                     Marital_Status(Separated or Divorced)
5.538188e-09                1.999411e+00            1.201402e+00
Mother_Negligence           Father_Negligence       Self_Estrangement
9.805836e-01                1.088317e+00            1.167837e+00
Isolation                   Normlessness            Meaninglessness
9.941947e-01                1.210831e+00            9.355597e-01
Drug_Use                    Metal                   Worshipping
1.372803e+00                1.145169e+00            1.172234e+00
Vicarious
7.104685e-01
```

We can also calculate confidence intervals for the odds ratios. To obtain confidence intervals of the parameters, we use the *confint()* function – just as we did for ordinary regression. We can also exponentiate these with the *exp()* function. To get the confidence intervals execute:

```
exp(confint(suicideModel))
```

This function computes the confidence intervals for the coefficients in the model (*confint(suicideModel)*) and then uses *exp()* to exponentiate them.

```
                                        2.5 %       97.5 %
(Intercept)                      9.065656e-15 0.0004577666
Age                              1.090546e+00 3.9372709912
Marital_StatusSeparated or Divorced 3.058015e-01 4.4898699168
Mother_Negligence                8.764185e-01 1.0837478703
Father_Negligence                9.932302e-01 1.2015111314
Self_Estrangement                1.035373e+00 1.3401633046
Isolation                        8.532270e-01 1.1548163908
Normlessness                     9.854717e-01 1.5210438866
Meaninglessness                  8.242768e-01 1.0500790337
```

```
Drug_Use                    1.132532e+00 1.7047218535
Metal                       9.596213e-01 1.3819317960
Worshipping                 9.084589e-01 1.5205816278
Vicarious                   4.714224e-01 1.0272917992
```

We can present these results in the following table:

	B	SE	95% CI for Odds Ratio		
			Lower	Odds Ratio	Upper
Constant	6.21	6.21			
Age	0.69*	0.32	1.09	2.00	3.94
Marital status	0.18	0.68	0.31	1.20	4.49
Mother negligence	−0.02	0.05	0.88	0.98	1.08
Father negligence	0.08*	0.05	0.99	1.09	1.20
Self-estrangement/ powerlessness	0.16*	0.06	1.04	1.17	1.34
Social isolation	−0.01	0.08	0.85	0.99	1.15
Normlessness	0.19*	0.11	0.99	1.21	1.52
Meaninglessness	−0.07	0.06	0.82	0.94	1.05
Drug use	0.32**	0.10	1.13	1.37	1.70
Metal	0.14	0.09	0.96	1.15	1.38
Worshipping	0.16*	0.13	0.91	1.17	1.52
Vicarious listening	−0.34	0.20	0.47	0.71	1.03

*p < .05, ** p < .01; one-tailed

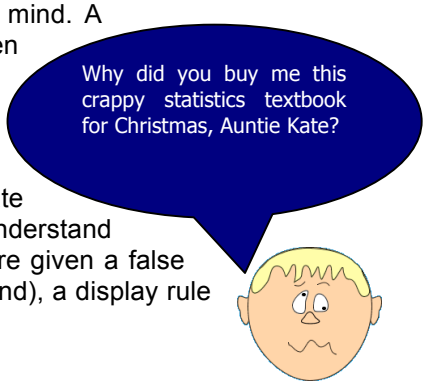
I've reported one-tailed significances (because Lacourse et al. do and it makes it easier to compare our results to Table 3 in their paper). We can conclude that listening to heavy metal did not significantly predict suicide risk in women (of course not; anyone I've ever met who likes metal does not conform to the stereotype). However, in case you're interested, listening to country music apparently does (Stack & Gundlach, 1992). The factors that did predict suicide risk were age (risk increased with age), father negligence (although this was significant only one-tailed, it showed that as negligence increased so did suicide risk), self-estrangement (basically low self-esteem predicted suicide risk, as you might expect), normlessness (again, only one-tailed), drug use (the more drugs used, the more likely a person was to be in the at-risk category), and worshipping (the more the person showed signs of worshipping bands, the more likely they were to be in the at-risk group).

The most significant predictor was drug use. So, this shows you that for girls, listening to metal was not a risk factor for suicide, but drug use was. To find out what happens for boys, you'll just have to read the article! This is scientific proof that metal isn't bad for your health, so download some Deathspell Omega and enjoy!

Smart Alex's solutions

Task 1

A psychologist was interested in whether children's understanding of display rules can be predicted from their age, and whether the child possesses a theory of mind. A display rule is a convention of displaying an appropriate emotion in a given situation. For example, if you receive a Christmas present that you don't like, the appropriate emotional display is to smile politely and say 'Thank you Auntie Kate, I've always wanted a rotting cabbage'. The inappropriate emotional display is to start crying and scream 'Why did you buy me a rotting cabbage, you selfish old bag?' Using appropriate display rules has been linked to having a theory of mind (the ability to understand what another person might be thinking). To test this theory, children were given a false belief task (a task used to measure whether someone has a theory of mind), a display rule



task (which they could either pass or fail) and their age in months was measured. The data are in **Display.dat**. Run a logistic regression to see whether possession of display rule understanding (did the child pass the test? yes/no) can be predicted from possession of a theory of mind (did the child pass the false belief task? yes/no), age in months and their interaction. ③

The main analysis

First of all we need to load the data:

```
displayData<-read.delim("Display.dat", header = TRUE)
```

We can have a look at the first six cases by executing:

```
 head(displayData)
```

```
  age fb   display
1  24 No      No
2  26 No      No
3  30 No      No
4  31 No      No
5  36 No      No
6  29 No     Yes
```

To carry out logistic regression, the data must be entered as for normal regression: they are arranged in three columns (one representing each variable). Looking at the *displayData* dataframe above, you should notice that both of the categorical variables have been loaded as factors and **R** will have specified numbers to represent the categories. For ease of interpretation, the outcome variable should be coded 1 (event occurred) and 0 (event did not occur); in this case, 1 represents having display rule understanding (Yes), and 0 represents an absence of display rule understanding (No). For the false belief task a similar coding has been used (0 = failed the false belief task (No), 1 = passed the false belief task (Yes)). Because **R** codes factors alphabetically, it will have coded No = 0 and Yes = 1 for both variables, which is what we want. Therefore, we do not need to relevel either of the factors for this example – woohoo!

OK, now we can get on with the analysis. I am going to do a hierarchical logistic regression. In the first model I am going to include the categorical predictor **fb** only, and then in the second model I will add the other predictor (**age**) and also the interaction between **fb** and **age** (**fb** × **age**, or *fb:age* in **R**). We can create these two models by executing:

```
displayModel.1 <- glm(display ~ fb, data = displayData, family = binomial())
displayModel.2 <- update(displayModel.1, .~. + age + fb:age)
```

To see the models that we have just generated we need to execute the *summary()* function (remembering to put the model name into the function):

```
summary(displayModel.1)
summary(displayModel.2)
```

```
Call:
glm(formula = display ~ fb, family = binomial(), data = displayData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8078  -0.6809   0.6589   0.6589   1.7751

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.3437     0.4584  -2.931  0.00338 **
fbYes         2.7608     0.6045   4.567 4.95e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 96.124  on 69  degrees of freedom
Residual deviance: 70.042  on 68  degrees of freedom
AIC: 74.042
```

```
Number of Fisher Scoring iterations: 4
```

Interpreting the output of devianceModel.1

The *null deviance* in the above output tells us about the model when only the constant is included (i.e. all predictor variables are omitted). This represents the fit of the model when the most basic model is fitted to the data. The *residual deviance* tells us about the model when the predictor **fb** has been added to the model. At this stage of the analysis the value of residual deviance should be less than the value when only the constant was included (null deviance) in the model (because lower values of $-2LL$ indicate that the model is predicting the outcome variable more accurately). When only the constant was included, $-2LL = 96.124$, but now **fb** has been included this value has been reduced to 70.042. This reduction tells us that the model is better at predicting display rule understanding than it was before **fb** was added.

The question of how much better the model predicts the outcome variable than the baseline model can be assessed using the *model chi-square statistic*, which measures the difference between the model as it currently stands and the model when only the constant was included. We can use **R** to automatically calculate the model chi-square and its significance by executing:

```
modelChi <- displayModel.1$null.deviance - displayModel.1$deviance
chidf <- displayModel.1$df.null - displayModel.1$df.residual
chisq.prob <- 1 - pchisq(modelChi, chidf)
```

We can then view the output of these commands by executing:

```
modelChi; chidf; chisq.prob
```

```
[1] 26.08266
[1] 1
[1] 3.271081e-07
```

The change in the amount of information explained by the model (26.08) is significant ($p < .0001$), so using false belief understanding as a predictor significantly improves our ability to predict display rule understanding, $\chi^2(1) = 26.08$, $p < .0001$.

The next part of the output is crucial because it tells us the estimates for the coefficients for the predictors included in the model. This section of the output gives us the coefficients and statistics for the variables that have been included in the model at this point (namely, **fb** and the constant). The crucial statistic is the *z*-statistic, which has a normal distribution and tells us whether the *b* coefficient for that predictor is significantly different from zero. If the coefficient is significantly different from zero then we can assume that the predictor is making a significant contribution to the prediction of the outcome. For these data it seems to indicate that false belief understanding, $b = 2.76$, $z = 4.57$, $p < .001$, is a significant predictor of display rule understanding (note that the significance of the *z*-statistic is less than .05).

Next, we want to calculate the various values of R^2, and we can do this by executing (remember that you need to have executed the function code from the book chapter first for this to work):

```
logisticPseudoR2s(displayModel.1)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.271
Cox and Snell R^2         0.311
Nagelkerke R^2            0.417
```

As you can see, all of the values of R^2 differ slightly, but they can be used as effect size measures for the model.

The final thing we need to look at is the *odds ratio* which was described in the book chapter. To calculate the change in odds that result from a unit change in the predictor for this example, we must first calculate the odds of a child having display rule understanding given that they don't have second-order false belief task understanding. We then calculate the odds of a child having display rule understanding given that they do have false belief understanding. Finally, we calculate the proportionate change in these two odds.

To calculate the first set of odds, we need to calculate the probability of a child having a display rule understanding given they failed the false belief task. The parameter coding at the beginning of the output told us that children who failed the false belief task were coded with a 0, so we can use this value in place of X. The value of b_1 has been estimated for us as 2.7607 and the coefficient for the constant can be taken from the same table and is −1.3437. We can calculate the odds as:

$$\begin{aligned} P(\text{event}Y) &= \frac{1}{1+e^{-(b_0+b_1X_1)}} \\ &= \frac{1}{1+e^{-[-1.3437+(2.7607\times 0)]}} \\ &= 0.2069 \end{aligned}$$

$$\begin{aligned} P(\text{no event}Y) &= 1 - P(\text{event}Y) \\ &= 1 - 0.2069 \\ &= 0.7931 \end{aligned}$$

$$\begin{aligned} \text{odds} &= \frac{0.2069}{0.7931} \\ &= 0.2609 \end{aligned}$$

Now, we calculate the same thing after the predictor variable has changed by one unit. In this case, because the predictor variable is dichotomous, we need to calculate the odds of a child passing the display rule task, given that they have passed the false belief task. So, the value of the false belief variable, X, is now 1 (rather than 0). The resulting calculations are:

$$\begin{aligned} P(\text{event}Y) &= \frac{1}{1+e^{-(b_0+b_1X_1)}} \\ &= \frac{1}{1+e^{-[-1.3437+(2.7607\times 1)]}} \\ &= 0.8049 \end{aligned}$$

$$\begin{aligned} P(\text{no event}Y) &= 1 - P(\text{event}Y) \\ &= 1 - 0.8049 \\ &= 0.1951 \end{aligned}$$

$$\begin{aligned} \text{odds} &= \frac{0.8049}{0.1951} \\ &= 4.1256 \end{aligned}$$

We now know the odds before and after a unit change in the predictor variable. It is now a simple matter to calculate the proportionate change in odds by dividing the odds after a unit change in the predictor by the odds before that change.

$$\begin{aligned} \Delta\text{odd} &= \frac{\text{odd after a unit change in the predictor}}{\text{original odd}} \\ &= \frac{4.1256}{0.2609} \\ &= 15.8129 \end{aligned}$$

If the value is greater than 1 then it indicates that as the predictor increases, the odds of the outcome occurring increase. Conversely, a value less than 1 indicates that as the predictor increases, the odds of the outcome occurring decrease. In this example, we can say that the odds of a child who has false belief understanding also having display rule understanding are 15 times higher than those of a child who does not have false belief understanding.

We can also get **R** to calculate the odds ratio as the exponential of the *b* coefficient for the predictor variables using the *exp()* function and by executing:

```
exp(displayModel.1$coefficients)
```

Executing this command will display the odds ratio for the predictors in the model:

```
(Intercept)       fbYes
  0.2608696  15.8125000
```

You should notice that the value of the proportionate change in odds that we calculated by hand above is the same as the value that **R** reports for exp(*b*) here (allowing for differences due to rounding).

We can also get **R** to calculate a confidence interval for the odds ratio. To obtain confidence intervals of the parameters, we use the *confint()* function – just as we did for ordinary regression. We can also exponentiate these with the *exp()* function. To get the confidence interval execute:

```
exp(confint(displayModel.1))
```

```
                2.5 %     97.5 %
(Intercept) 0.0963625  0.6012293
fbYes       5.1415334 56.1604505
```

The way to interpret this confidence interval is to say that if we ran 100 experiments and calculated confidence intervals for the value of exp(*b*), then these intervals would encompass the actual value of exp(*b*) in the population (rather than the sample) on 95 occasions. So, in this case, we can be fairly confident that the population value of exp(*b*) lies between 5.14 and 56.16. However, there is a 5% chance that a sample could give a confidence interval that 'misses' the true value. What's more, because both values are greater than 1 we can also be confident that the relationship between **fb** and display rule understanding found in this sample is true of the whole population of children. If we had found that the confidence interval ranged from less than 1 to more than 1, then this would limit the generalizability of our findings because the odds ratio in the population could indicate either a positive (odds ratio > 1) or negative (odds ratio < 1) relationship.

The output for model 2 (below) shows what happens to the model when our new predictor (**age**) and the interaction (**fb** × **age**) are added. The effect of adding these to the model is to reduce the –2*LL* to 67.63 (a reduction of 2.412 from model 1), suggesting that including them in the model has not improved our ability to predict whether a child will have display rule understanding or not. In addition, we can see that the AIC is slightly higher in model 2 (75.63) than model 1 (74.04), indicating that model 1 is the better model.

```
Call:
glm(formula = display ~ fb + age + fb:age, family = binomial(),
    data = displayData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0339  -0.6236   0.4875   0.6909   1.9968

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.94817    1.59648  -1.847   0.0648 .
fbYes        2.85775    2.10523   1.357   0.1746
age          0.04404    0.04060   1.085   0.2781
fbYes:age   -0.01669    0.04757  -0.351   0.7256
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 96.124  on 69  degrees of freedom
Residual deviance: 67.634  on 66  degrees of freedom
AIC: 75.634

Number of Fisher Scoring iterations: 4
```

We can again compare the models by finding the difference in the deviance statistics. This difference is chi-square distributed. We can find this difference in two ways. First, we can subtract one deviance from the other as we did before. An easier method, though, is to use the *anova()* function. The *anova()* function has the advantage that it also calculates the degrees of freedom for us. If we do the calculations manually we can use the same commands as before, except that rather than using the *null.deviance* and *df.null* variables, we use the *deviance* and *df.residual* variables for the two models we're comparing – in each case we subtract model 2 from model 1:

```
modelChi <- displayModel.1$deviance - displayModel.2$deviance
chidf <- displayModel.1$df.residual - displayModel.2$df.residual
chisq.prob <- 1 - pchisq(modelChi, chidf)

modelChi; chidf; chisq.prob
```

```
[1] 2.407671
[1] 2
[1] 0.3000412
```

You should find that the difference between the models (*modelChi*) is 2.407671, with 2 degrees of freedom (*chidf*), and a *p*-value (*chisq.prob*) of .3000412. As this value is greater than .05, we can conclude that model 2 (with **age** and **fb** × **age** added as predictors) is not a significant improvement over model 1 (which included only **fb** as a predictor).

We can do the same with the *anova()* function. Remember that with this function we simply list the models in the order in which we want to compare them. Therefore, to compare our two models we would execute:

```
anova(displayModel.1, displayModel.2)
```

```
Analysis of Deviance Table

Model 1: display ~ fb
Model 2: display ~ fb + age + fb:age
  Resid. Df Resid. Dev Df Deviance
1        68     70.042
2        66     67.634  2   2.4077
```

The coefficients part of the output now contains all three predictors, and something very interesting has happened: understanding of false beliefs (**fb**) no longer significantly predicts display rules. In addition, **age** appears not to make a significant contribution to the prediction of display rules, and neither does the interaction between age and false beliefs. How can it be that false beliefs no longer predict display rule understanding, and neither does age or the interaction, yet the ability of the model to predict display rule understanding has improved slightly?

Let's again compute R^2 for model 2 (remember to execute the function code first if you haven't done so already):

```
logisticPseudoR2s(displayModel.2)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.296
Cox and Snell R^2         0.334
Nagelkerke R^2            0.448
```

We can next calculate the odds ratio by executing:

```
exp(displayModel.2$coefficients)
```

Executing this command will display the odds ratio for the predictors in the model:

```
(Intercept)  fbYes         age       fbYes:age
 0.05243577 17.42231132  1.04502296  0.98344567
```

We can also calculate confidence intervals for the odds ratios, as we did for model 1:

```
exp(confint(displayModel.2))
```

```
                2.5 %      97.5 %
(Intercept) 0.001759117    1.142108
fbYes       0.293461449 1292.673012
age         0.961033191    1.136442
fbYes:age   0.893710835    1.083236
```

If we examine the values of the odds ratio for both **fb** and **age** it is clear that they both potentially have a positive relationship to display rule understanding (i.e. as they increase by a unit, the odds of understanding improve). However, the confidence intervals for these values cross 1, which indicates that the direction of this relationship may be unstable in the population as a whole (i.e. the value of the odds ratio in our sample may be quite different than the value if we had data from the entire population).

You may be tempted to use this final model to say that, the previous finding that understanding of false beliefs plays a role in display rule understanding is incorrect. This would be a dangerous conclusion to make, and if you have read the section on multicollinearity in the book you'll see why.

Predicted probabilities

Let's take a look at the predicted probabilities. We can use the *head()* function again just to look at the first few cases, by executing:

```
head(displayData[, c("display", "fb", "age", "predicted.probabilities")])
```

This command uses displays the first six cases, and we have selected a subset of variables from the *eelData* dataframe.

```
  display fb age       predicted.probabilities
1  No      No  24                0.2068966
2  No      No  26                0.2068966
3  No      No  30                0.2068966
4  No      No  31                0.2068966
5  No      No  36                0.2068966
6 Yes      No  29                0.2068966
```

The output above shows the values of the predicted probabilities as well as the initial data (you will need to see more of the output to interpret the all of the results; to see the whole output, just run the command without putting *head* at the beginning).

We found from the model that the only significant predictor of display rule understanding was false belief understanding. This could have a value of either 1 (pass the false belief task) or 0 (fail the false belief task). These values tells us that when a child doesn't possess second-order false belief understanding (**fb** = 0, No), there is a probability of .2069 that they will pass the display rule task, approximately a 21% chance (1 out of 5 children). However, if the child does pass the false belief task (**fb** = 1, yes), there is a probability of .8049 that they will pass the display rule task, an 80.5% chance (4 out of 5 children). Consider that a probability of 0 indicates no chance of the child passing the display rule task, and a probability of 1 indicates that the child will definitely pass the display rule task. Therefore, the values obtained provide strong evidence for the role of false belief understanding as a prerequisite for display rule understanding.

Assuming we are content that the model is accurate and that false belief understanding has some substantive significance, then we could conclude that false belief understanding is the single best predictor of display rule understanding. Furthermore, age and the interaction of age and false belief understanding do not significantly predict display rule understanding.

This conclusion is fine in itself, but to be sure that the model is a good one, it is important to examine the residuals.

Interpreting residuals

We saw in the previous chapter that the main purpose of examining residuals in any regression is to (1) isolate points for which the model fits poorly, and (2) isolate points that exert an undue influence on the model. To assess the former we examine the residuals, especially the studentized residual, standardized residual and deviance statistics. To assess the latter we use influence statistics such as Cook's distance, DFBeta and leverage statistics.

If you have saved your residuals in the dataframe then you could look at them by executing something like:

```
displayData[, c("leverage", "studentized.residuals", "dfbeta")]
```

This command will print the leverage, studentized residuals and DFBeta values for the model.

The basic residual statistics for this example (leverage, studentized residuals and DFBeta values) are pretty good (see the output below): note that all cases have DFBetas less than 1, and leverage statistics are very close to the calculated expected value of 0.018. All in all, this means that there are no influential cases having an effect on the model. The studentized residuals all have values between –2 and 2 and so there seems to be very little here to concern us.

```
leverage           studentized.residuals dfbeta.(Intercept) dfbeta.fbYes
1  0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
2  0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
3  0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
4  0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
5  0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
6  0.03448276             1.8132781       1.565055e-01 -1.565055e-01
7  0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
8  0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
9  0.03448276             1.8132781       1.565055e-01 -1.565055e-01
10 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
11 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
12 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
13 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
14 0.02439024            -1.8361335      -8.294434e-18 -1.140460e-01
15 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
16 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
17 0.03448276             1.8132781       1.565055e-01 -1.565055e-01
18 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
19 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
20 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
21 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
22 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
23 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
24 0.02439024            -1.8361335       8.294434e-18 -1.140460e-01
25 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
26 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
27 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
28 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
29 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
30 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
31 0.03448276             1.8132781       1.565055e-01 -1.565055e-01
32 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
33 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
34 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
35 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
36 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
37 0.02439024            -1.8361335       3.317774e-17 -1.140460e-01
38 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
39 0.03448276             1.8132781       1.565055e-01 -1.565055e-01
40 0.02439024            -1.8361335       4.976661e-17 -1.140460e-01
41 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
42 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
43 0.02439024             0.6634679      -5.391382e-17  4.156533e-02
44 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
45 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
46 0.02439024            -1.8361335       1.244165e-16 -1.140460e-01
47 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
48 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
49 0.03448276            -0.6876918      -6.003071e-02  6.003071e-02
50 0.02439024            -1.8361335       1.244165e-16 -1.140460e-01
51 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
52 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
53 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
54 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
55 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
56 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
57 0.02439024            -1.8361335       1.492998e-16 -1.140460e-01
58 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
59 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
60 0.03448276             1.8132781       1.565055e-01 -1.565055e-01
61 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
62 0.02439024             0.6634679      -4.147217e-17  4.156533e-02
```

```
63 0.02439024            0.6634679      -3.317774e-17  4.156533e-02
64 0.02439024            0.6634679      -3.317774e-17  4.156533e-02
65 0.02439024            0.6634679      -3.317774e-17  4.156533e-02
66 0.02439024            0.6634679      -3.317774e-17  4.156533e-02
67 0.02439024            0.6634679      -3.317774e-17  4.156533e-02
68 0.02439024           -1.8361335       1.492998e-16 -1.140460e-01
69 0.02439024            0.6634679      -3.317774e-17  4.156533e-02
70 0.02439024            0.6634679      -3.317774e-17  4.156533e-02
```

You should note that these residuals are slightly unusual because they are based on a single predictor that is categorical. This is why there isn't a lot of variability in the values of the residuals. Also, if substantial outliers or influential cases had been isolated, you are not justified in eliminating these cases to make the model fit better. Instead these cases should be inspected closely to try to isolate a good reason why they were unusual. It might simply be an error in inputting data, or it could be that the case was one which had a special reason for being unusual: for example, the child had found it hard to pay attention to the false belief task and you had noted this at the time of the experiment. In such a case, you may have good reason to exclude the case and duly note the reasons why.

Task 2

- Recent research has shown that lecturers are among the most stressed workers. A researcher wanted to know exactly what it was about being a lecturer that created this stress and subsequent burnout. She took 467 lecturers and administered several questionnaires to them that measured: **Burnout** (burnt out or not), **Perceived Control** (high score = low perceived control), **Coping Style** (high score = low ability to cope with stress), **Stress from Teaching** (high score = teaching creates a lot of stress for the person), **Stress from Research** (high score = research creates a lot of stress for the person) and **Stress from Providing Pastoral Care** (high score = providing pastoral care creates a lot of stress for the person). The outcome of interest was burnout, and Cooper, Sloan, and Williams's (1988) model of stress indicates that perceived control and coping style are important predictors of this variable. The remaining predictors were measured to see the unique contribution of different aspects of a lecturer's work to their burnout. Can you help her out by conducting a logistic regression to see which factor predict burnout? The data are in **Burnout.dat**.

Test

Obviously we need to begin by loading the data:

```
burnoutData<-read.delim("Burnout.dat", header = TRUE)
```

We can see the first six cases of data by executing:

```
head(burnoutData)
```

```
   burnout        loc      cope     teaching research pastoral
1 Not Burnt Out  7.647059  9.160305 32.72727 87.50000 31.48148
2 Not Burnt Out  6.470588 12.977099 52.72727 66.66667 68.51852
3 Not Burnt Out  8.823529  9.160305 49.09091 60.41667 53.70370
4 Not Burnt Out 20.000000  9.160305 52.72727 62.50000 50.00000
5 Not Burnt Out  6.470588 19.083969 43.63636 79.16667 40.74074
6 Not Burnt Out  7.058824 16.030534 38.18182 52.08333 48.14815
```

Looking at the data above, we can see that the categorical variable **burnout** has two categories, *Not Burnt Out* and *Burnt Out*, and **R** has encoded it as a factor. If you think back to the eel example in the chapter, you will remember that **R** encodes factors in alphabetical order, which doesn't always make the most sense for our data analysis. In this particular example, this is a problem because *Burnt Out* will be category 1 and *Not Burnt Out* category 2. In other words, *Burnt Out* will be the baseline category, but it makes more sense for *Not Burnt Out* to be the baseline category. Not to worry, though, we can set *Not Burnt Out* to be the baseline by executing:

```
burnoutData$burnout<-relevel(burnoutData$burnout, "Not Burnt Out")
```

OK, now we can get stuck into the analysis. The analysis should be done hierarchically because Cooper et al.'s model indicates that perceived control and coping style are important predictors of burnout. So, these variables should be entered in the first block. The second block should contain all other variables because we don't know anything much about their predictive ability.

We can do this by using the *glm()* function to create two models as we did in the chapter. To create the first model we can execute:

```
burnoutModel.1 <- glm(burnout ~ loc + cope, data = burnoutData, family = binomial())
```

This command creates a model called *burnoutModel.1* in which **burnout** is predicted from **loc** (perceived control) and **cope** (coping ability). Similarly, we can create the second model by executing:

```
burnoutModel.2 <- update(burnoutModel.1, .~. + teaching + research + pastoral)
```

This command creates a model called *burnoutModel.2* in which **burnout** is predicted from **loc**, **cope**, **teaching**, **research** and **pastoral**.

To see the models that we have just generated we need to execute the *summary()* function (remembering to put the model name into the function):

```
summary(burnoutModel.1)
summary(burnoutModel.2)
```

```
Call:
glm(formula = burnout ~ loc + cope, family = binomial(), data = burnoutData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9217  -0.5163  -0.3730   0.1273   2.0848

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.484493   0.379458 -11.818  < 2e-16 ***
loc          0.061080   0.010915   5.596 2.19e-08 ***
cope         0.082714   0.009369   8.829  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 530.11  on 466  degrees of freedom
Residual deviance: 364.18  on 464  degrees of freedom
AIC: 370.18

Number of Fisher Scoring iterations: 5
```

The output above shows the model summary for the model 1 (*burnoutModel.1*). We will begin by looking at the summary statistics about the model. The overall fit of the model is assessed using the deviance statistic. Remember that larger values of the deviance statistic indicate poorly fitting statistical models. **R** provides two deviance statistics: the null deviance, and the residual deviance. The null deviance is the deviance of the model that contains no predictors, other than the constant. The residual deviance is the deviance for the model.

At this stage of the analysis the value of the deviance for the model should be less than the value for the null model (when only the constant was included in the model), because lower values of $-2LL$ indicate that the model is predicting the outcome variable more accurately. For the null model $-2LL = 530.11$, but when **cope** and **loc** have been included this value has been reduced to 364.18. This reduction tells us that the model is better at predicting burnout than it was before **cope** and **loc** were added.

The question of how much better the model predicts the outcome variable can be assessed using the *model chi-square statistic*, which measures the difference between the model as it

currently stands and the model when only the constant was included. We can use **R** to automatically calculate the model chi-square and its significance by executing:

```
modelChi <- burnoutModel.1$null.deviance - burnoutModel.1$deviance
chidf <- burnoutModel.1$df.null - burnoutModel.1$df.residual
chisq.prob <- 1 - pchisq(modelChi, chidf)
```

We can then view the output of these commands by executing:

```
modelChi; chidf; chisq.prob
```

```
[1] 165.9277
[1] 2
[1] 0
```

The change in the amount of information explained by the model (165.93) is significant (p < .0001), so using coping ability and perceived control as predictors significantly improves our ability to predict whether someone will be burnt out or not, $\chi^2(2) = 165.93$, $p < .0001$.

Next, we consider the coefficients. This part is crucial because it tells us the estimates for the coefficients for the predictors included in the model. This section of the output gives us the coefficients and statistics for the variables that have been included in the model at this point (namely **loc**, **cope** and the constant). The crucial statistic is the z-statistic, which has a normal distribution and tells us whether the b coefficient for that predictor is significantly different from zero. If the coefficient is significantly different from zero then we can assume that the predictor is making a significant contribution to the prediction of the outcome. For these data it seems to indicate that coping ability, $b = 0.08$, z = 8.83, $p < .001$, and perceived control, $b = 0.06$, z = 5.60, $p < .001$, are both significant predictors of burnout (note that the significance of the z-statistics is less than .05 in both cases).

Next, we want to calculate the various values of R^2, we can do this by executing (remember that you need to have executed the function code from the book chapter first):

```
 logisticPseudoR2s(burnoutModel.1)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.313
Cox and Snell R^2         0.299
Nagelkerke R^2            0.441
```

As you can see, all of the values of R^2 differ slightly, but they can be used as effect size measures for the model. The model can account for 31% of the variance in prediction whether someone is burnt out or not (depending on what measure of R^2 you use).

We can next calculate the odds ratio by executing:

```
exp(burnoutModel.1$coefficients)
```

```
(Intercept)         loc        cope
 0.01128261  1.06298389  1.08623164
```

The value of the odds ratio for **loc** indicates that as the level of perceived control increases (high score = low perceived control), then the odds of being burnt out also increase (because the odds ratio is greater than 1). The odds ratio for **cope** indicates that if the level of coping ability increases (high score = low ability to cope with stress), then the odds of being burnt out also increase (because the odds ratio is also greater than 1).

We can also calculate confidence intervals for the odds ratios. To obtain confidence intervals of the parameters, we use the *confint()* function – just as we did for ordinary regression. We can also exponentiate these with the *exp()* function. To get the confidence intervals execute:

```
exp(confint(burnoutModel.1))
```

```
                2.5 %     97.5 %
(Intercept) 0.005160721 0.02292526
loc         1.041229885 1.08691181
cope        1.067210914 1.10722003
```

The confidence interval for **loc** ranges from 1.04 to 1.08, so we can be very confident that the value of the odds ratio in the population lies somewhere between these two values. What's more, because both values are greater than 1 we can also be confident that the relationship between **loc** (perceived coping ability) and burnout found in this sample is true of the whole population of lecturers. The confidence interval for **cope** ranges from 1.06 to 1.10, so we can be very confident that the value of the odds ratio in the population lies somewhere between these two values. In addition, because both values are greater than 1 we can be confident that the relationship between **cope** and burnout found in this sample is true of the whole population of lecturers. If we had found that the confidence interval ranged from less than 1 to more than 1, then this would limit the generalizability of our findings because the odds ratio in the population could indicate either a positive (odds ratio > 1) or negative (odds ratio < 1) relationship.

The output for model 2 (below) shows what happens to the model when our new predictors are added. The effect of adding these to the model is to reduce the $-2LL$ to 321.20 (a reduction of 42.98 from model 1, suggesting that including the other predictors in the model has improved our ability to predict whether a lecturer will be burnt out or not.

```
Call:
glm(formula = burnout ~ loc + cope + teaching + research + pastoral,
    family = binomial(), data = burnoutData)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.41592  -0.48290  -0.28690   0.02966   2.63636

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.43993    1.08565  -4.090 4.32e-05 ***
loc          0.11079    0.01494   7.414 1.23e-13 ***
cope         0.14234    0.01639   8.684  < 2e-16 ***
teaching    -0.11216    0.01977  -5.673 1.40e-08 ***
research     0.01931    0.01036   1.863 0.062421 .
pastoral     0.04517    0.01310   3.449 0.000563 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 530.11  on 466  degrees of freedom
Residual deviance: 321.20  on 461  degrees of freedom
AIC: 333.2

Number of Fisher Scoring iterations: 6
```

We can again compare model 2 with the baseline model (the model before any predictors are added to the model) by finding the difference in the deviance statistics by executing:

```
modelChi <- burnoutModel.2$null.deviance - burnoutModel.2$deviance
chidf <- burnoutModel.2$df.null - burnoutModel.2$df.residual
chisq.prob <- 1 - pchisq(modelChi, chidf)

modelChi; chidf; chisq.prob
```

```
[1] 208.9086
[1] 5
[1] 0
```

You should find that the difference between the models (*modelChi*) is 208.9086, with 5 degrees of freedom (*chidf*), and a *p*-value (*chisq.prob*) given as 0. As this value is less than .05, we can conclude that the overall fit of the model is significant, $\chi^2(5) = 208.91$, $p < .0001$.

The coefficients part of the output now contains all three predictors, and we can see that all the predictors significantly predict burn out except for **research**.

Let's again compute R^2 for model 2 (remember to execute the function code first if you haven't done so already):

```
logisticPseudoR2s(burnoutModel.2)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.394
Cox and Snell R^2         0.361
```

```
Nagelkerke R^2            0.531
```

Overall, the final model accounts for 36.1–53.1% of the variance in burnout (depending on which measure of R^2 you use.

We can next calculate the odds ratio by executing:

```
exp(burnoutModel.2$coefficients)
```

Executing this command will display the odds ratio for the predictors in the model:

```
(Intercept)   loc         cope       teaching    research    pastoral
 0.01179680  1.11715594  1.15296414  0.89389904  1.01949919  1.04620942
```

We can also calculate confidence intervals for the odds ratios, as we did for model 1:

```
exp(confint(burnoutModel.2))
```

```
              2.5 %       97.5 %
(Intercept) 0.001317788 0.09419003
loc         1.086274965 1.15212014
cope        1.118430575 1.19286786
teaching    0.858532732 0.92793154
research    0.999115252 1.04068582
pastoral    1.020119629 1.07403586
```

In terms of the individual predictors we could report:

	B (SE)	95% CI for Odds Ratio		
		Lower	Odds Ratio	Upper
Step 1				
Constant	–4.48** (0.38)			
Perceived Control	0.06** (0.01)	1.04	1.06	1.09
Coping Style	0.08** (0.01)	1.07	1.09	1.11
Final				
Constant	–4.44** (1.09)			
Perceived Control	0.11** (0.01)	1.08	1.12	1.15
Coping Style	0.14** (0.02)	1.12	1.15	1.19
Teaching Stress	–0.11** (0.02)	0.86	0.89	0.93
Research	0.02 (0.01)	1.00	1.02	1.04
Pastoral Stress	0.05* (0.01)	1.02	1.05	1.07

Note: R^2 = .36 (Cox and Snell), .53 (Nagelkerke). Model $\chi^2(5) = 208.91$, $p < .001$. * $p < .01$, ** $p < .001$.

It seems as though burnout is significantly predicted by perceived control, coping style (as predicted by Cooper), stress from teaching and stress from giving pastoral care. The odds ratio and direction of the beta values tell us that, for perceived control, coping ability and pastoral care, the relationships are positive. That is (and look back to the question to see the direction of these scales, i.e. what a high score represents), poor perceived control, poor ability to cope with stress, and stress from giving pastoral care all predict burnout. However, for teaching, the relationship is the opposite way around: stress from teaching appears to be a positive thing as it predicts not becoming burnt out!

Task 3

- A health psychologist interested in research into HIV wanted to know the factors that influenced condom use with a new partner (relationship less than 1 month old). The outcome measure was whether a condom was used (**use**: condom used = 1, not used = 0). The predictor variables were mainly scales from the Condom Attitude Scale (CAS) by Sacco, Levine, Reed, and Thompson (1991): **gender** (gender of the person); **safety** (relationship safety, measured out of 5, indicates the degree to which the person views this relationship as 'safe' from sexually transmitted disease); **sexexp** (sexual experience, measured out of 10, indicates the degree to which previous experience influences attitudes towards condom use); **previous** (a measure not from the CAS, this variable measures whether or not the couple used a condom in their previous encounter: 1 = condom used, 0 = not used, 2 = no previous encounter with this partner); **selfcon** (self-control, measured out of 9, indicates the degree of self-control that a person has when it comes to condom use, i.e. whether they get carried away with the heat of the moment, or exert control); **perceive** (perceived risk, measured out of 6, indicates the degree to which the person feels at risk from unprotected sex). Previous research (Sacco, Rickman, Thompson, Levine, & Reed, 1993) has shown that gender, relationship safety and perceived risk predict condom use. Carry out an appropriate analysis to verify these previous findings, and to test whether self-control, previous usage and sexual experience can predict any of the remaining variance in condom use. (1) Interpret all important parts of the **R** output. (2) How reliable is the final model? (3) What are the probabilities that participants 12, 53 and 75 will use a condom? (4) A female who used a condom in her previous encounter with her new partner scores 2 on all variables except perceived risk (for which she scores 6). Use the model to estimate the probability that she will use a condom in her next encounter. Data are in the file **condom.dat**.

Remember to load the data first:

```
condomData<-read.delim("condom.dat", header = TRUE)
```

Let's have a look at first six cases of data by executing:

```
head(condomData)
```

```
participant safety       use     gender  sexexp  previous selfcon perceive
1         5      3 Unprotected Female      5 No Condom       5        4
2         6      1 Unprotected   Male      3 No Condom       2        2
3         9      0 Unprotected Female      2 No Condom       3        0
4        13      3 Unprotected   Male      3 No Condom       4        4
5        14      2 Unprotected Female      3 No Condom       6        3
6        18      0 Unprotected Female      8 No Condom       5        1
```

We need to relevel the categorical variables in the dataframe:

```
condomData$use<-relevel(condomData$use, "Unprotected")
condomData$gender<-relevel(condomData$gender, "Male")
```

For the categorical variable **previous** it makes sense to specify the baseline category as "*no condom*". We can do this by executing:

```
condomData$previous<-relevel(condomData$previous, "No Condom")
```

The correct analysis was to run a hierarchical logistic regression entering **perceive**, **safety** and **gender** in the first block and **previous**, **selfcon** and **sexexp** in a second. We can create these two models by executing:

```
condomModel.1 <- glm(use ~ perceive + safety + gender, data = condomData, family =
binomial())
condomModel.2 <- update(condomModel.1, .~. + previous + selfcon + sexexp)
```

We can then see the output of the two models by executing:

```
summary(condomModel.1)
summary(condomModel.2)
```

Let's first look at *condomModel.1*. The output of this model is shown below:

```
Call:
glm(formula = use ~ perceive + safety + gender, family = binomial(),
    data = condomData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0523  -1.0255   0.3228   0.8832   2.3567

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   2.1594     0.6349   3.401 0.000671 ***
perceive     -0.9402     0.2230  -4.217 2.48e-05 ***
safety        0.4641     0.2178   2.131 0.033099 *
genderMale    0.3167     0.4963   0.638 0.523361
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 136.66  on 99  degrees of freedom
Residual deviance: 105.77  on 96  degrees of freedom
AIC: 113.77
```

The first thing to note is that $-2LL$ has dropped from 136.66 (*null deviance*) to 105.77 (*residual deviance*), which is a change of 30.89 (which is the value given by the *model chi-square*). This value tells us about the model as a whole. We can test the significance of this change in $-2LL$ by executing:

```
modelChi <- condomModel.1$null.deviance - condomModel.1$deviance
chidf <- condomModel.1$df.null - condomModel.1$df.residual
chisq.prob <- 1 - pchisq(modelChi, chidf)
modelChi; chidf; chisq.prob
```

```
[1] 30.89248
[1] 3
[1] 8.955305e-07
```

The output above tells us that the change in the amount of information explained by the model is significant ($\chi^2(3) = 30.89$, $p < .0001$), so using perceived risk, relationship safety and gender as predictors significantly improves our ability to predict condom use.

Next, we consider the coefficients. This part is crucial because it tells us the estimates for the coefficients for the predictors included in the model. This section of the output gives us the coefficients and statistics for the variables that have been included in the model at this point (namely **perceive**, **safety**, **gender** and the constant). The crucial statistic is the *z*-statistic which has a normal distribution and tells us whether the *b* coefficient for that predictor is significantly different from zero. If the coefficient is significantly different from zero then we can assume that the predictor is making a significant contribution to the prediction of the outcome. For these data it seems to indicate that the values for perceived risk, $b = -0.94$, $z = -4.22$, $p < .001$, and relationship safty, $b = 0.46$, $z = 2.13$, $p < .05$, are both significant predictors of condom use (note that the significance of the *z*-statistics is less than .05 in both cases). However, gender was not a significant predictor of whether someone was likely to use a condom, $b = 0.32$, $z = 0.64$, $p > .05$.

Next, we want to calculate the various values of R^2, and we can do this by executing (remember that you need to have executed the function code from the book chapter first):

```
logisticPseudoR2s(condomModel.1)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.226
Cox and Snell R^2         0.266
Nagelkerke R^2            0.357
```

As you can see, all of the values of R^2 differ slightly, but they can be used as effect size measures for the model. The model can account for 22.6–35.7% of the variance in predicting whether someone will use a condom or not (depending on what measure of R^2 you use).

We can next calculate the odds ratio by executing:

```
exp(condomModel.1$coefficients)
```

```
 (Intercept)     perceive      safety     genderFemale
  0.08407051     2.56047130   0.62867995   1.37260745
```

We can also calculate confidence intervals for the odds ratios. To obtain confidence intervals of the parameters, we use the *confint()* function – just as we did for ordinary regression. We can also exponentiate these with the *exp()* function. To get the confidence intervals execute:

```
exp(confint(condomModel.1))
```

```
                2.5 %    97.5 %
(Intercept)  0.01682132 0.3309217
perceive     1.73455558 4.2126668
safety       0.38809459 0.9277043
genderFemale 0.52183556 3.6998237
```

The values for perceived risk (odds ratio 2.56, confidence interval from 1.73 to 4.21) indicate that if the value of perceived risk goes up by 1, then the odds of using a condom also increase (because the odds ratio is greater than 1). The confidence interval for this value ranges from 1.73 to 4.21 so we can be very confident that the value of the odds ratio in the population lies somewhere between these two values. What's more, because both values are greater than 1 we can also be confident that the relationship between perceived risk and condom use found in this sample is true of the whole population. In short, as perceived risk increase by 1, people are just over twice as likely to use a condom.

The values for relationship safety (odds ratio 0.63, confidence interval from 0.39 to 0.93) indicate that if the relationship safety increases by one point, then the odds of using a condom decrease (because the odds ratio is less than 1). The confidence interval for this value ranges from 0.39 to 0.93 so we can be very confident that the value of the odds ratio in the population lies somewhere between these two values. In addition, because both values are less than 1 we can be confident that the relationship between relationship safety and condom use found in this sample is true of the whole population. In short, as relationship safety increases by one unit, subjects are about 1.6 times less likely to use a condom.

The values for gender (odds ratio 1.37, confidence interval from 0.52 to 3.70) indicate that as gender changes from 0 (male) to 1 (female), then the odds of using a condom increase (because the odds ratio is greater than 1). However, the confidence interval for this value crosses 1, which limits the generalizability of our findings because the odds ratio in other samples (and hence the population) could indicate either a positive (odds ratio > 1) or negative (odds ratio < 1) relationship. Therefore, gender is not a reliable predictor of condom use.

The output below shows what happens to the model when our new predictors are added (previous use, self-control and sexual experience). This part of the output describes block 2, which is just the model described in block 1 but with a new predictors added. So, we begin with the model that we had in block 1 and we then add **previous**, **selfcon** and **sexexp** to it. The effect of adding these predictors to the model is to reduce the $-2LL$ to 87.971 (a reduction of 48.69 from the original model and an additional reduction of 17.799 from the reduction caused by model 1).

```
Call:
glm(formula = use ~ perceive + safety + gender + previous + selfcon +
    sexexp, family = binomial(), data = condomData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2237  -0.6905  -0.2072   0.6244   1.9748

Coefficients:
                                Estimate Std. Error z value Pr(>|z|)
(Intercept)                    -4.959739   1.146497  -4.326 1.52e-05 ***
perceive                        0.949088   0.236972   4.005 6.20e-05 ***
safety                         -0.482460   0.236033  -2.044  0.04095 *
genderFemale                    0.002656   0.572823   0.005  0.99630
previousCondom used             1.087196   0.551952   1.970  0.04887 *
previousFirst Time with partner -0.016615   1.399907  -0.012  0.99053
selfcon                         0.347626   0.126842   2.741  0.00613 **
```

```
sexexp                          0.180423   0.111586   1.617  0.10590
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 136.663  on 99  degrees of freedom
Residual deviance:  87.971  on 92  degrees of freedom
AIC: 103.97
```

We can test whether the improvement from model 1 to model 2 is significant by executing:

```
modelChi <- condomModel.1$deviance - condomModel.2$deviance
chidf <- condomModel.1$df.residual - condomModel.2$df.residual
chisq.prob <- 1 - pchisq(modelChi, chidf)
 modelChi; chidf; chisq.prob
```

```
[1] 17.79902
[1] 4
[1] 0.001350844
```

The output above tells us that the additional improvement of block 2 is significant ($\chi^2(4) = 17.80$, $p < .01$), which tells us that including these three new predictors in the model has significantly improved our ability to predict condom use.

```
Coefficients:
                                 Estimate Std. Error z value Pr(>|z|)
(Intercept)                     -4.959739   1.146497  -4.326 1.52e-05 ***
perceive                         0.949088   0.236972   4.005 6.20e-05 ***
safety                          -0.482460   0.236033  -2.044  0.04095 *
genderFemale                     0.002656   0.572823   0.005  0.99630
previousCondom used              1.087196   0.551952   1.970  0.04887 *
previousFirst Time with partner -0.016615   1.399907  -0.012  0.99053
selfcon                          0.347626   0.126842   2.741  0.00613 **
sexexp                           0.180423   0.111586   1.617  0.10590
```

The coefficients section of the output now contains all predictors. This part of the output represents the details of the final model. The significance values of the *b* coefficient for each predictor indicate that both perceived risk ($b = 0.94$, $z = 4.00$, $p < .001$) and relationship safety ($b = -0.48$, $z = -2.04$, $p < .001$) still significantly predict condom use and, as in block 1, gender does not ($b = 0.00$, $z = 0.01$, $p > .05$). We can now look at the new predictors to see which of these has some predictive power.

Previous use has been split into two components (according to whatever contrasts were specified for this variable). From the output we can see that **previousCondom used** compares the condom used group against the other two, and **previousFirst Time with partner** compares the base category of first time with partner against the other two categories. Therefore, we can tell that previous use is not a significant predictor of condom use when it is the first time with a partner compared to when it is not the first time ($b = -0.02$, $z = -0.01$, $p > .05$). However, when we compare the condom used category to the other categories we find that using a condom on the previous occasion does predict use on the current occasion ($b = 1.09$, $z = 1.97$, $p < .05$).

Of the other new predictors, we find that self-control predicts condom use ($b = 0.35$, $z = 2.74$, $p < .01$) but sexual experience does not ($b = 0.18$, $z = 1.62$, $p > .05$).

Let's again compute R^2 for model 2 (remember to execute the function code first if you haven't done so already in this session):

```
logisticPseudoR2s(condomModel.2)
```

```
Pseudo R^2 for logistic regression
Hosmer and Lemeshow R^2   0.356
Cox and Snell R^2         0.385
Nagelkerke R^2            0.517
```

Overall, the final model accounts for 35.6–51.7% of the variance in predicting condom use (depending on which measure R^2 you use).

We can next calculate the odds ratio by executing:

```
exp(condomModel.2$coefficients)
```

Executing this command will display the odds ratio for the predictors in the model:

```
(Intercept)         perceive                           safety
0.007014758         2.583353254                     0.617263292
gender:Female        previous:Condom used   previous:First Time with partner
1.002659308         2.965946499                     0.983522066
selfcon             sexexp
1.415702224         1.197724363
```

We can also calculate confidence intervals for the odds ratios, as we did for model 1:

```
exp(confint(condomModel.2))
```

```
                                 2.5 %          97.5 %
(Intercept)                 0.0005824489  0.05487106
perceive                    1.6995027286  4.36242303
safety                      0.3671545834  0.94143728
genderFemale                0.3208849083  3.10396280
previousCondom used         1.0303140867  9.17513490
previousFirst Time with partner 0.0361032447 13.94704684
selfcon                     1.1188404702  1.85122675
sexexp                      0.9702216719  1.50978237
```

Looking at the output for the odds ratio and confidence intervals above, we can see that the values for perceived risk (odds ratio 2.58, confidence interval from 1.70 to 4.36) indicate that if the value of perceived risk goes up by 1, then the odds of using a condom also increase. What's more, because the confidence interval doesn't cross 1 we can also be confident that the relationship between perceived risk and condom use found in this sample is true of the whole population. As perceived risk increases by 1, people are just over twice as likely to use a condom.

The values for relationship safety (odds ratio 0.62, confidence interval from 0.37 to 0.94) indicate that if the relationship safety decreases by one point, then the odds of using a condom decrease. The confidence interval does not cross 1 so we can be confident that the relationship between relationship safety and condom use found in this sample is true of the whole population. As relationship safety increases by one unit, subjects are about 1.6 times less likely to use a condom.

The values for gender (odds ratio 1.00, confidence interval from 0.32 to 3.10) indicate that as gender changes from 0 (male) to 1 (female), then the odds of using a condom do not change (because the odds ratio is equal to 1). The confidence interval crosses 1, therefore gender is not a reliable predictor of condom use.

The values for previous condom use (odds ratio 2.97, confidence interval from 1.03 to 9.18) indicate that if the value of previous usage goes up by 1 (i.e. changes from not having used one or being the first time to having used one), then the odds of using a condom also increase. What's more, because the confidence interval doesn't cross 1 we can also be confident that this relationship is true in the whole population. If someone used a condom on their previous encounter with this partner (compared to if they didn't use one, or if it is their first time) then they are three times more likely to use a condom. For previous first time with partner the odds ratio (0.98, confidence interval from 0.04 to 13.95) indicates that if the value of previous usage goes up by 1 (i.e. changes from not having used one or having used one to being their first time with this partner), then the odds of using a condom do not change (because the value is very nearly equal to 1). What's more, because the confidence interval crosses 1 we can tell that this is not a reliable predictor of condom use.

The values for self-control (odds ratio 1.42, confidence interval from 1.12, 1.85) indicate that if self-control increases by one point, then the odds of using a condom increase also. The confidence interval does not cross 1 so we can be confident that the relationship between relationship safety and condom use found in this sample is true of the whole population. As self-control increases by one unit, subjects are about 1.4 times more likely to use a condom.

The values for sexual experience (odds ratio 1.20, confidence interval from 0.97, 1.51) indicate that as sexual experience increases by one unit, then the odds of using a condom increase slightly. However, the confidence interval crosses 1, therefore sexual experience is not a reliable predictor of condom use.

How reliable is the final model?

Multicollinearity can affect the parameters of a regression model. Logistic regression is equally prone to the biasing effect of collinearity, and it is essential to test for collinearity following a logistic regression analysis. We can get the VIF and tolerance as we did in Chapter 7 by entering the model name into the *vif()* function from the *car* package. Execute:

```
vif(condomModel.2)
1/vif(condomModel.2)
```

The first block gives the VIF values and the second the tolerance (which is simply the reciprocal of the VIF).

```
           GVIF    Df     GVIF^(1/(2*Df))
perceive 1.794738  1        1.339678
safety   1.783765  1        1.335577
gender   1.169653  1        1.081505
previous 1.083281  2        1.020200
selfcon  1.060989  1        1.030043
sexexp   1.120416  1        1.058497

           GVIF      Df    GVIF^(1/(2*Df))
perceive 0.5571845 1.0       0.7464479
safety   0.5606119 1.0       0.7487402
gender   0.8549542 1.0       0.9246373
previous 0.9231214 0.5       0.9802000
selfcon  0.9425165 1.0       0.9708329
sexexp   0.8925260 1.0       0.9447359
```

Looking at the tolerance values, we can see that for all variables the tolerance values are close to 1 and are much larger than the cut-off point of 0.1 below which Menard (1995) suggests there may be a serious collinearity problem. Myers (1990) also suggests that a VIF value greater than 10 is cause for concern and in these data the values are all less than this criterion.

We can also test for linearity of the logit. In this example we have four continuous variables, therefore we have to check that each one is linearly related to the log of the outcome variable (**use**). I mentioned in the book chapter that to test this assumption we need to run the logistic regression but include predictors that are the interaction between each predictor and the log of itself (Hosmer & Lemeshow, 1989). We need to create the log of each of the variables, by using the *log()* function:

```
condomData$logsafety<-log(condomData$safety +1)
condomData$logsexexp<-log(condomData$sexexp +1)
condomData$logselfcon<-log(condomData$selfcon + 1)
condomData$logperceive<-log(condomData$perceive +1)
```

To test the assumption we need to redo the analysis exactly the same as before except that we should put all variables in a single block (i.e., we don't need to do it hierarchically), and we also need to put in three new interaction terms of each predictor and their logs. We create the model by executing:

```
condomTest.1 <- glm(use ~ safety + sexexp + selfcon + perceive + safety:logsafety +
sexexp:logsexexp + selfcon:logselfcon + perceive:logperceive, data = condomData,
family=binomial())
```

We then use the *summary()* function to display the model:

```
summary(condomTest.1)
```

```
Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept)          -33.9873    17.3118  -1.963   0.0496 *
safety               -20.9723    15.4100  -1.361   0.1735
sexexp                -1.3946     1.0968  -1.272   0.2035
selfcon               -0.3946     1.1061  -0.357   0.7213
perceive              42.2504    25.0871   1.684   0.0922 .
safety:logsafety      10.0270     7.3288   1.368   0.1713
sexexp:logsexexp       0.6537     0.4449   1.469   0.1417
selfcon:logselfcon     0.2976     0.4708   0.632   0.5273
perceive:logperceive -17.5402    10.4679  -1.676   0.0938 .
```

The above table shows the part of the output that tests the assumption. We're interested only in whether the interaction terms are significant. Any interaction that is significant indicates that the main effect has violated the assumption of linearity of the logit. All four interactions have significance values (the values in the column *Pr(>|z|)*) greater than .05, indicating that the assumption of linearity of the logit has been met for **safety**, **sexexp**, **selfcon** and **perceive**.

Residuals should be checked for influential cases and outliers. As with linear regression, it is possible to calculate residuals. These residual variables can then be examined to see how well the model fits the observed data. The commands to obtain residuals are the same as those we encountered for linear regression. To obtain residuals, we can use the *resid()* function and include the model name within it.

Fitted values for logistic regression are a little different from linear regression. The fitted values are the predicted probabilities of *Y* occurring given the values of each predictor for a given participant. We can also calculate a predicted group membership, based on the most likely outcome for each person based on the model. The group memberships are based on the predicted probabilities, and I will explain these values in more detail when we consider how to interpret the residuals. Predicted probabilities are obtained with the *fitted()* function (again, we simply supply the model name to the function).

As with ordinary regression, then, we can add these casewise diagnostic variables to our dataframe by creating new variables to contain them and then using the various functions we encountered in section 7.9.2 to populate these variables with the appropriate values. For example, as a basic set of diagnostic statistics we might execute:

```
condomData$predicted.probabilities<-fitted(condomModel.2)
condomData$standardized.residuals<-rstandard(condomModel.2)
condomData$studentized.residuals<-rstudent(condomModel.2)
condomData$dfbeta<-dfbeta(condomModel.2)
condomData$dffit<-dffits(condomModel.2)
```

You might want to save the file after creating these variables by executing:

```
write.table(condomData, "condom With Diagnostics.dat", sep = "\t", row.names = FALSE)
```

Interpreting residuals

If you have saved your residuals in the dataframe then you could look at them by executing something like:

```
condomData[, c("leverage", "studentized.residuals", "dfbeta")]
```

This command will print the leverage, studentized residuals and DFBeta values for model. I haven't pasted the output here as there is quite a lot of it and it would take up a lot of space!

The basic residual statistics for this example (leverage, studentized residuals and DFBeta values) are pretty good: note that all cases have DFBetas less than 1, and leverage statistics are very close to the calculated expected value of 0.018. All in all, this means that there are no influential cases having an effect on the model. The studentized residuals all have values between –2 and +2 and so there seems to be very little here to concern us.

What are the probabilities that participants 12, 53 and 75 will use a condom?

We want the predicted probabilities of condom use for participants, 12, 53 and 75 only. We can get these by executing the same command as we used for calculating the predicted probabilities before, but adding *c(12,53,75)* before the comma in the command. Therefore, we would execute:

```
 (condomData[c(12,53,75),    c("use",    "safety",    "sexexp","selfcon",    "perceive",
"previous", "gender", "predicted.probabilities")])

     use        safety sexexp selfcon perceive    previous gender    predicted.probabilities
12 Unprotected      3      1       1        6   No Condom Female               0.4546541
53 Unprotected      5      3       7        6 Condom used   Male               0.9156252
75 Condom Used      2      2       4        3 Condom used Female               0.4412250
```

The output above tells us that the probablility that participant 12 will use a condom is 45%, the probability that participant 53 will use a condom is 92% and the probability that participant 75 will use a condom is 44%.

A female, who used a condom in her previous encounter with her new partner, scores 2 on all variables except perceived risk (for which she scores 6). Use the model to estimate the probability that she will use a condom in her next encounter.
Step 1: Logistic regression equation:

$$P(Y) = \frac{1}{1 + e^{-Z}}$$

where $Z = b_0 + b_1X_1 + b_2X_2 + \ldots + b_nX_n$,

Step 2: Use the values of *b* from the **R** output of model 2 to construct the following table:

Variable	b_i	X_i	$b_i X_i$
gender	0.0027	1	0.0027
safety	−0.4825	2	−0.965
sexexp	0.1804	2	0.3608
previousCondom used	1.0872	1	1.0872
previousFirst Time with partner	−0.0167	0	0
selfcon	0.3476	2	0.6952
perceive	0.9491	6	5.6946

Step 3: Place the values of $b_i X_i$ into the equation for *z* (remembering to include the constant):

$$z = -4.9597 + 0.0027 - 0.965 + 0.3608 + 1.0872 + 0 + 0.6952 + 5.6934$$
$$= 1.9146$$

Step 4: Replace this value of *z* into the logistic regression equation (NB: you can calculate '*e* to the power of –1.9146' in **R** by typing *exp(-1.9146)*):

$$PY = 11 + e - Z$$
$$= 11 + e - 1.9146$$
$$= 11 + 0.1474008$$
$$= 0.8715349$$

Therefore, there is a 87% chance that she will use a condom on her next encounter.