

CHAPTER 21

Dichotomous Predicted Variable

Contents

21.1.	Multiple Metric Predictors	622
21.1.1	The model and implementation in JAGS	622
21.1.2	Example: Height, weight, and gender	626
21.2.	Interpreting the Regression Coefficients	629
21.2.1	Log odds	629
21.2.2	When there are few 1's or 0's in the data	631
21.2.3	Correlated predictors	632
21.2.4	Interaction of metric predictors	633
21.3.	Robust Logistic Regression	635
21.4.	Nominal Predictors	636
21.4.1	Single group	638
21.4.2	Multiple groups	641
21.4.2.1	Example: Baseball again	641
21.4.2.2	The model	642
21.4.2.3	Results	644
21.5.	Exercises	646

*Fortune and Favor make fickle decrees, it's
Heads or it's tails with no middle degrees.
Flippant commandments decreed by law gods, have
Reasons so rare they have minus log odds.¹*

This chapter considers data structures that consist of a dichotomous predicted variable. The early chapters of the book were focused on this type of data, but now we reframe the analyses in terms of the generalized linear model. Data structures of the type considered in this chapter are often encountered in real research. For example, we might want to predict whether a person in a demographic study would be male or female based on their height and weight. Or we might want to predict whether or not a person will vote, based on their annual income. Or we might want to predict whether a baseball batter will get a hit, based on what their primary fielding position is.

The traditional treatment of these sorts of data structure is called “logistic regression.” In Bayesian software it is easy to generalize the traditional models so they are robust

¹ This chapter is about logistic regression, and one of the concepts is called “log odds,” explained in Section 21.2.1. I was fortunate to rhyme “log odds” with “law gods” and then work backwards to their names, Fortune and Favor.

to outliers, allow different variances within levels of a nominal predictor, and have hierarchical structure to share information across levels or factors as appropriate.

In the context of the generalized linear model (GLM) introduced in Chapter 15, this chapter's situation involves an inverse-link function that is logistic along with a Bernoulli distribution for describing noise in the data, as indicated in the second row of Table 15.2 (p. 443). For a reminder of how this chapter's combination of predicted and predictor variables relates to other combinations, see Table 15.3 (p. 444).

21.1. MULTIPLE METRIC PREDICTORS

We begin by considering a situation with multiple metric predictors, because this case makes it easiest to visualize the concepts of logistic regression. Suppose we measure the height, weight, and gender (male or female) of a sample of full-grown adults. From everyday experience, it seems plausible that a tall, heavy person is more likely to be male than a short, light person. But exactly how predictive of gender is height or weight?

Some representative data are shown in [Figure 21.1](#). The data are fictitious but realistic, generated by an accurate model of a large population survey (Brainard & Burmester, 1992). The data are plotted as 1's or 0's, with gender arbitrarily coded as male = 1 and female = 0. All 0's are located on the bottom plane and all 1's are located on the top plane. You can see that the 1's tend to be at larger values of height and weight, while the 0's tend to be at smaller values of height and weight. (A “top down” view is shown in [Figure 21.4](#), p. 628.)

21.1.1. The model and implementation in JAGS

How can we describe the rise in the probability of 1's as height and weight increase in [Figure 21.1](#)? We will use a logistic function of a linear combination of the predictors. This type of model was introduced in Section 15.3.1.1 (p. 436), and in Figure 15.10 (p. 443). The idea is that a linear combination of metric predictors is mapped to a probability value via the logistic function, and the predicted 0's and 1's are Bernoulli distributed around the probability. Restated formally:

$$\begin{aligned}\mu &= \text{logistic}(\beta_0 + \beta_1 x_1 + \beta_2 x_2) \\ \gamma &\sim \text{Bernoulli}(\mu)\end{aligned}$$

where

$$\text{logistic}(x) = \frac{1}{(1 + \exp(-x))}$$

The intercept and slope parameters (i.e., β_0 , β_1 , and β_2) are given prior distributions in the usual ways, although the interpretation of logistic regression coefficients requires careful thought, as will be explained in [Section 21.2.1](#).

$$y \sim \text{dbern}(m), m = \text{logistic}(0.018x_1 + 0.7x_2 - 50)$$

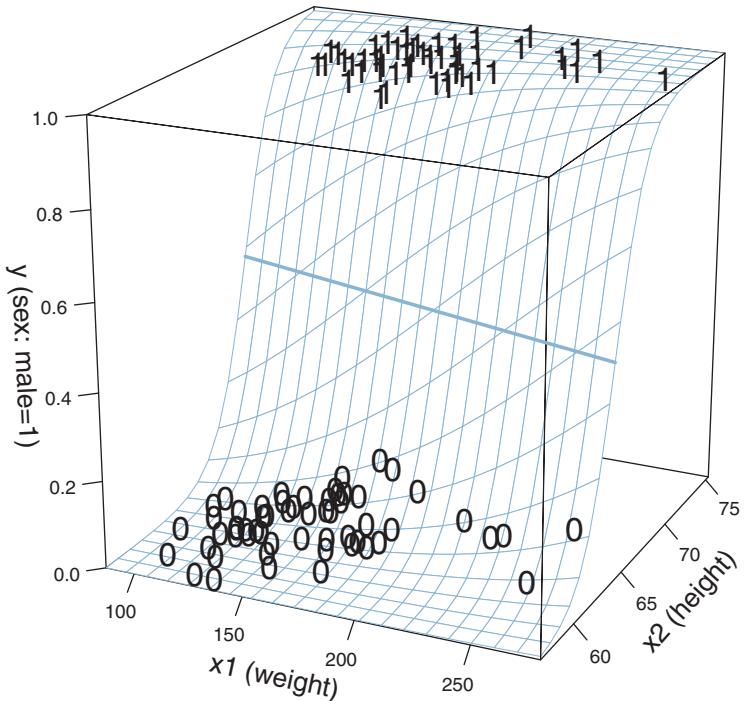


Figure 21.1 Data show gender (arbitrarily coded as male = 1, female = 0) as a function of weight (in pounds) and height (in inches). All 0's are located on the bottom plane of the cube, and all 1's are located on the top plane of the cube. Logistic surface shows maximum-likelihood estimate. Heavy line shows 50% level contour.

A diagram of the model is presented in [Figure 21.2](#). At the bottom of the diagram, each dichotomous y_i value comes from a Bernoulli distribution with a “bias” of μ_i . (Recall that I refer to the parameter in the Bernoulli distribution as the bias, regardless of its value. Thus, a coin with a bias value of 0.5 is fair.) The μ value is determined as the logistic function of the linear combination of predictors. Finally, the intercept and slope parameters are given conventional normal priors at the top of the diagram.

It can be useful to compare the diagram for multiple logistic regression (in [Figure 21.2](#)) with the diagram for robust multiple linear regression in [Figure 18.4](#) (p. 515). The linear core of the model is the same in both diagrams. What differs across diagrams is the bottom of the diagram that describes y , because y is on different types of scales in the two models. In the present application ([Figure 21.2](#)), y is dichotomous and therefore is described as coming from a Bernoulli distribution. In the earlier application ([Figure 18.4](#),

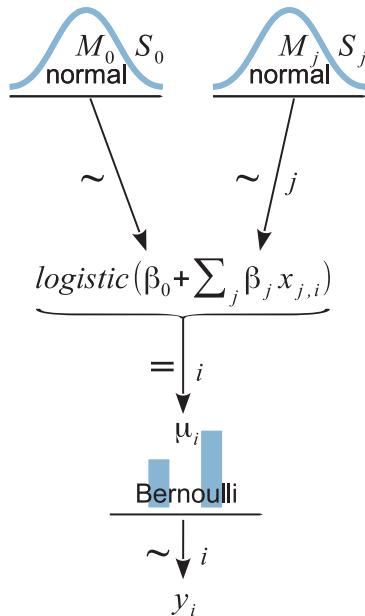


Figure 21.2 Dependency diagram for multiple logistic regression. Compare with the diagram for robust multiple linear regression in Figure 18.4 (p. 515).

p. 515), γ was metric and therefore was described as coming from a t distribution (for robustness against outliers).

It is straight forward to compose JAGS (or Stan) code for the model in Figure 21.2. Before getting to the JAGS code itself, there are a couple more preliminary details to explain. First, as we did for linear regression, we will try to reduce autocorrelation in the MCMC chains by standardizing the data. This is done merely to improve the efficiency of the MCMC process, not out of logical necessity. The y values must be 0's and 1's and therefore are not standardized, but the predictor values are metric and are standardized. Recall from Equation 17.1 (p. 485) that we denote the standardized value of x as $z = (x - \bar{x}) / s_x$, where \bar{x} is the mean of x and s_x is the standard deviation of x . And, recall from Equation 15.15 (p. 439) that the inverse logistic function is called the logit function. We standardize the data and let JAGS find credible parameter values, denoted ζ_0 and ζ_j , for the standardized data. We then transform the standardized parameter values to the original scale, as follows:

$$\begin{aligned}\text{logit}(\mu) &= \zeta_0 + \sum_j \zeta_j z_j \\ &= \zeta_0 + \sum_j \zeta_j \frac{x_j - \bar{x}_j}{s_{xj}}\end{aligned}$$

$$= \underbrace{\zeta_0 - \sum_j \frac{\zeta_j}{s_{x_j}} \bar{x}_j}_{\beta_0} + \sum_j \underbrace{\frac{\zeta_j}{s_{x_j}}}_{\beta_j} x_j \quad (21.1)$$

The transformation indicated by the underbraces in [Equation 21.1](#) is applied at every step in the MCMC chain.

One more detail before getting to the JAGS code. In JAGS, the logistic function is called `ilogit`, for inverse logit. Recall from Section 15.3.1.1 (p. 436) that the inverse link function goes from the linear combination of predictors to the predicted tendency of the data, whereas the link function goes from the predicted tendency of the data to the linear combination of predictors. In this application, the link function is the logit. The inverse link function is, therefore, the inverse logit, abbreviated in JAGS as `ilogit`. The inverse link function is also, of course, the logistic. The `ilogit` notation helps distinguish the logistic function from the logistic distribution. We will not be using the logistic distribution, but just for your information, the logistic distribution is the probability density function which has a cumulative probability given by the logistic function. In other words, the logistic distribution is to the logistic function as the normal distribution is to the cumulative normal function in Figure 15.8 (p. 440). The logistic distribution is similar to the normal distribution but with heavier tails. Again, we will not be using the logistic distribution, and the point of this paragraph has been to explain JAGS' use of the term `ilogit` for the logistic function.

The JAGS code for the logistic regression model in [Figure 21.2](#) is presented below. It begins with a data block that standardizes the data. Then the `model` block expresses the dependency arrows in [Figure 21.2](#), and finally the parameters are transformed to the original scale using [Equation 21.1](#):

```
# Standardize the data:
data {
  for ( j in 1:Nx ) {
    xm[j] <- mean(x[,j])
    xsd[j] <- sd(x[,j])
    for ( i in 1:Ntotal ) {
      zx[i,j] <- ( x[i,j] - xm[j] ) / xsd[j]
    }
  }
}
# Specify the model for standardized data:
model {
  for ( i in 1:Ntotal ) {
    # In JAGS, ilogit is logistic:
    y[i] ~ dbern( ilogit( zbeta0 + sum( zbeta[1:Nx] * zx[i,1:Nx] ) ) )
  }
}
```

```

# Priors vague on standardized scale:
zbeta0 ~ dnorm( 0 , 1/2^2 )
for ( j in 1:Nx ) {
  zbeta[j] ~ dnorm( 0 , 1/2^2 )
}
# Transform to original scale:
beta[1:Nx] <- zbeta[1:Nx] / xsd[1:Nx]
beta0 <- zbeta0 - sum( zbeta[1:Nx] * xm[1:Nx] / xsd[1:Nx] )
}

```

The line in the JAGS code that specified the Bernoulli distribution did not use a separate line for specifying μ_i because we did not want to record the μ_i values. If we wanted to, we would instead explicitly create μ_i as follows:

```

y[i] ~ dbern( mu[i] )
mu[i] <- ilogit( zbeta0 + sum( zbeta[1:Nx] * zx[i,1:Nx] ) )

```

The complete program is in the file named `Jags-Ydich-XmetMulti-Mlogistic.R`, and the high-level script is the file named `Jags-Ydich-XmetMulti-Mlogistic-Example.R`.

21.1.2. Example: Height, weight, and gender

Consider again the data in [Figure 21.4](#), which show the gender (male = 1, female = 0), height (in inches), and weight (in pounds) for 110 adults. We would like to predict gender from height and weight. We will first consider the results of using a single predictor, weight, and then consider the results of using both predictors.

[Figure 21.3](#) shows the results of predicting gender from weight alone. The data are plotted as points that fall only at 0 or 1 on the y -axis. Superimposed on the data are logistic curves that have parameter values from various steps in the MCMC chain. The spread of the logistic curves indicates the uncertainty of the estimate; the steepness of the logistic curves indicates the magnitude of the regression coefficient. The 50% probability threshold is marked by arrows that drop down from the logistic curve to the x -axis, near a weight of approximately 160 pounds. The threshold is the x value at which $\mu = 0.5$, which is $x = -\beta_0/\beta_1$. You can see that the logistic appears to have a clear positive (nonzero) rise as weight increases, suggesting that weight is indeed informative for predicting gender. But the rise is gentle, not steep: There is no sharp threshold weight below which most people are female and above which most people are male.

The lower panels of [Figure 21.3](#) show the marginal posterior distribution on the parameters. In particular, the slope coefficient, β_1 , has a mode larger than 0.03 and a 95% HDI that is well above zero (presumably by enough to exclude at least some non-zero ROPE). A later section will discuss how to interpret the numerical value of the regression coefficient.

Now we consider the results when using two predictors, namely both height and weight, as shown in [Figure 21.4](#). The data are plotted as 1's and 0's with x_1 (weight)

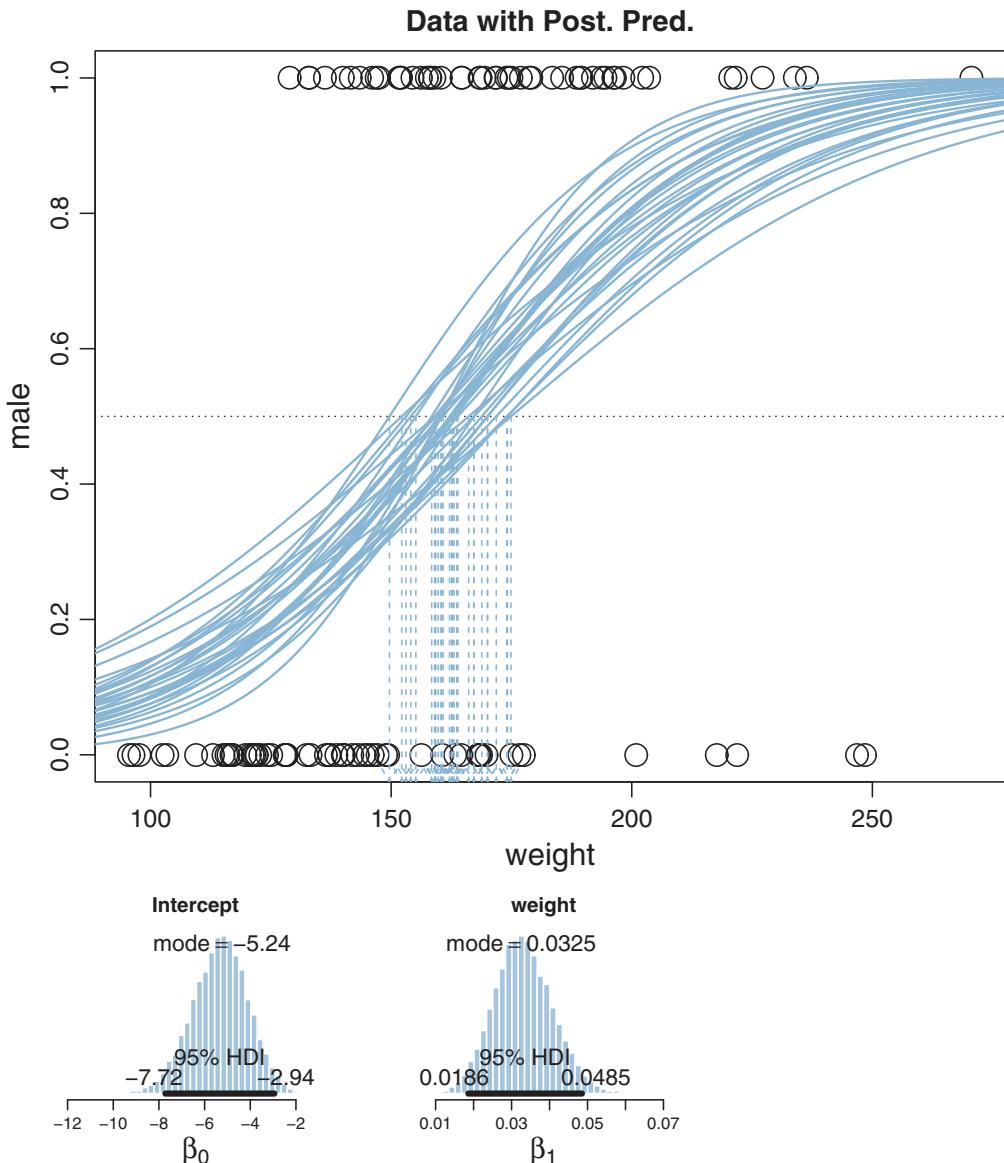


Figure 21.3 Predicting gender (arbitrarily coded as male = 1, female = 0) as a function of weight (in pounds), using logistic regression. Upper panel: Data are indicated by dots. Logistic curves are a random sample from the MCMC posterior. Descending arrows point to threshold weights at which the probability of male is 50%. Lower panels: Marginal posterior distribution on intercept and slope.

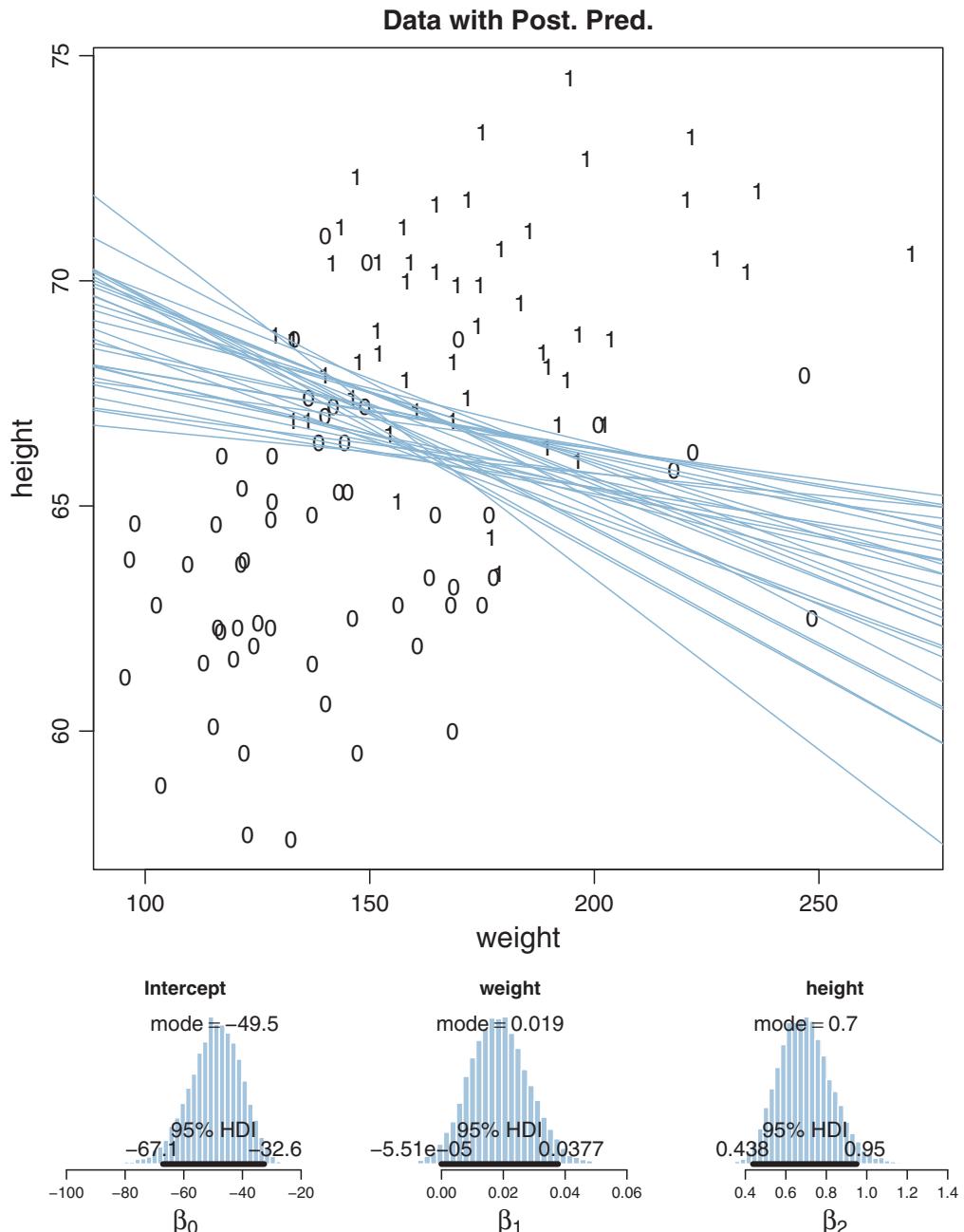


Figure 21.4 Predicting gender (arbitrarily coded as male = 1, female = 0) as a function of weight (in pounds) and height (in inches), using logistic regression. Upper panel: Data are indicated by 0's and 1's. Lines show a random sample from the MCMC posterior of thresholds at which the probability of male is 50%. Lower panels: Marginal posterior distribution on intercept and slopes.

on the horizontal axis and x_2 (height) on the vertical axis. You can see from the scatter plot of points that weight and height are positively correlated. Superimposed on the data are credible level contours at which $p(\text{male}) = 50\%$. Look back at [Figure 21.1](#) (p. 623) for a picture of how a 50% level contour falls on a logistic surface. The 50% level contour is the set of x_1, x_2 values for which $\mu = 0.5$, which is satisfied by $x_2 = (-\beta_0/\beta_2) + (-\beta_1/\beta_2)x_1$. On one side of the level contour there is less than 50% chance of being male, and on the other side of the level contour there is greater than 50% chance of being male, according to the model (not necessarily in reality). The spread of the credible level contours indicates the uncertainty in the parameter estimates. The perpendicular to the level contour indicates the direction in which probability changes the fastest. The angle of the level contours in [Figure 21.4](#) suggests that the probability of being male increases rapidly as height goes up, but the probability of being male increases only a little as weight goes up. This interpretation is confirmed by the lower panels of [Figure 21.4](#), which show the marginal posterior on the regression coefficients. In particular, the regression coefficient on weight has a modal value less than 0.02 and its 95% HDI essentially touches zero.

The regression coefficient on weight is smaller when height is included in the regression than when height is not included (compare [Figure 21.4](#) with [Figure 21.3](#)). Why would this be? Perhaps you anticipated the answer: Because the predictors are correlated, and it's really height that's doing most of the predictive work in this case. When weight is the only predictor included, it has some predictive power because weight correlates with height, and height predicts gender. But when height is included as a second predictor, we find that the independent predictiveness of weight is relatively small. This issue of interpreting correlated predictors is the same here, in the context of logistic regression, as was discussed at length in Section 18.1.1 (p. 510) in the context of linear regression.

21.2. INTERPRETING THE REGRESSION COEFFICIENTS

In this section, I'll discuss how to interpret the parameters in logistic regression. The first subsection explains how to interpret the numerical magnitude of the slope coefficients in terms of "log odds." The next subsection shows how data with relatively few 1's or 0's can yield ambiguity in the parameter estimates. Then an example with strongly correlated predictors reveals tradeoffs in slope coefficients. Finally, I briefly describe the meaning of multiplicative interaction for logistic regression.

21.2.1. Log odds

When the logistic regression formula is written using the logit function, we have $\text{logit}(\mu) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$. The formula implies that whenever x_1 goes up by 1 unit (on the x_1 scale), then $\text{logit}(\mu)$ goes up by an amount β_1 . And whenever x_2 goes up by 1 unit (on the x_2 scale), then $\text{logit}(\mu)$ goes up by an amount β_2 . Thus,

the regression coefficients are telling us about increases in $\text{logit}(\mu)$. To understand the regression coefficients, we need to understand $\text{logit}(\mu)$.

The logit function is the inverse of the logistic function. Formally, for $0 < \mu < 1$, $\text{logit}(\mu) = \log(\mu/(1 - \mu))$, where the logarithm is the natural logarithm, that is, the inverse of the exponential function. It's easy to verify through algebra that this expression for the logit really does make it the inverse of the logistic: If $x = \text{logit}(\mu) = \log(\mu/(1 - \mu))$, then $\mu = \text{logistic}(x) = 1/(1 + \exp(-x))$ and vice versa. Now, in applications to logistic regression, μ is the probability that $y = 1$, and therefore we can write $\text{logit}(\mu) = \text{logit}(p(y=1)) = \log(p(y=1)/(1 - p(y=1))) = \log(p(y=1)/p(y=0))$. The ratio, $p(y=1)/p(y=0)$, is called the odds of outcome 1 to outcome 0, and therefore $\text{logit}(\mu)$ is the log odds of outcome 1 to outcome 0.

Combining the previous two paragraphs, we can say that the regression coefficients are telling us about increases in the log odds. Let's consider a numerical example. Rounding the modal values in Figure 21.4, suppose that $\beta_0 = -50.0$, $\beta_1 = 0.02$, and $\beta_2 = 0.70$.

- Consider a hypothetical person who weighs 160 pounds, that is, $x_1 = 160$. If that person were 63 inches tall, then the predicted probability of being male is $\text{logistic}(\beta_0 + \beta_1 x_1 + \beta_2 x_2) = \text{logistic}(-50.0 + 0.02 \cdot 160 + 0.70 \cdot 63) = 0.063$. That probability is a log odds of $\log(0.063/(1 - 0.063)) = -2.70$. Notice the negative value of the log odds, which indicates the probability is less than 50%. If that person were 1 inch taller, at 64 inches, then the predicted probability of being male is 0.119, which has a log odds of -2.00 . Thus, when x_2 was increased by 1 unit, the probability went up 0.056 (from 0.063 to 0.119), and the log odds increased 0.70 (from -2.70 to -2.00) which is exactly β_2 .
- Again consider a hypothetical person who weighs 160 pounds, that is, $x_1 = 160$. Now consider an increase in height from 67 to 68 inches. If the person were 67 inches tall, then the predicted probability of being male is $\text{logistic}(\beta_0 + \beta_1 x_1 + \beta_2 x_2) = \text{logistic}(-50.0 + 0.02 \cdot 160 + 0.70 \cdot 67) = 0.525$. That probability is a log odds of $\log(0.525/(1 - 0.525)) = 0.10$. Notice the positive value of the log odds, which indicates the probability is greater than 50%. If that person were 1 inch taller, at 68 inches, then the predicted probability of being male is 0.690, which has a log odds of 0.80 . Thus, when x_2 was increased by 1 unit, the probability went up 0.165 (from 0.525 to 0.690), and the log odds increased 0.70 (from 0.10 to 0.80) which is exactly β_2 .

The two numerical examples show that an increase of 1 unit of x_j increases the log odds by β_j . But a constant increase in log odds does not mean a constant increase in probability. In the first numerical example, the increase in probability was 0.056, but in the second numerical example, the increase in probability was 0.165.

Thus, a regression coefficient in logistic regression indicates how much a 1 unit change of the predictor increases the log odds of outcome 1. A regression coefficient

of 0.5 corresponds to a rate of probability change of about 12.5 percentage points per x -unit at the threshold x value. A regression coefficient of 1.0 corresponds to a rate of probability change of about 25.0 percentage points per x -unit at the threshold x value. When x is much larger or smaller than the threshold x value, the rate of change in probability is smaller, even though the rate of change in log odds is constant.

21.2.2. When there are few 1's or 0's in the data

In logistic regression, you can think of the parameters as describing the boundary between the 0's and the 1's. If there are many 0's and 1's, then the estimate of the boundary parameters can be fairly accurate. But if there are few 0's or few 1's, the boundary can be difficult to identify very accurately, even if there are many data points overall.

In the data of [Figure 21.1](#), involving gender (male, female) as a function of weight and height, there were approximately 50% 1's (males) and 50% 0's (females). Because there were lots of 0's and 1's, the boundary between them could be estimated relatively accurately. But many realistic data sets have only a small percentage of 0's or 1's. For example, suppose we are studying the incidence of heart attacks predicted by blood pressure. We measure the systolic blood pressure at the annual check-ups of a random sample of people, and then record whether or not they had a heart attack during the following year. We code not having a heart attack as 0 and having a heart attack as 1. Presumably the incidents of heart attack will be very few, and the data will have very few 1's. This dearth of 1's in the data will make it difficult to estimate the intercept and slope of the logistic function with much accuracy.

[Figure 21.5](#) shows an example. For both panels, the x values of the data are identical, random values from a standardized normal distribution. For both panels, the y values were generated randomly from a Bernoulli distribution with bias given by a logistic function with the same slope, $\beta_1 = 1$. What differs between panels is where the threshold of the logistic falls relative to the mean of the x values. For the left panel, the threshold is at 3 (i.e., $\beta_0 = -3$), whereas for the right panel, the threshold is at 0. You can see that the data in the left panel have relatively few 1's (in fact, only about 7% 1's), whereas the data in the right panel have about 50% 1's. The right panel is analogous to the case of gender as a function of weight. The left panel is analogous to the case of heart attack as a function of blood pressure.

You can see in [Figure 21.5](#) that the estimate of the slope (and of the intercept) is more certain in the right panel than in the left panel. The 95% HDI on the slope, β_1 , is much wider in the left panel than in the right panel, and you can see that the logistic curves in the left panel have greater variation in steepness than the logistic curves in the right panel. The analogous statements hold true for the intercept parameter.

Thus, if you are doing an experimental study and you can manipulate the x values, you will want to select x values that yield about equal numbers of 0's and 1's for the y values overall. If you are doing an observational study, such that you cannot control

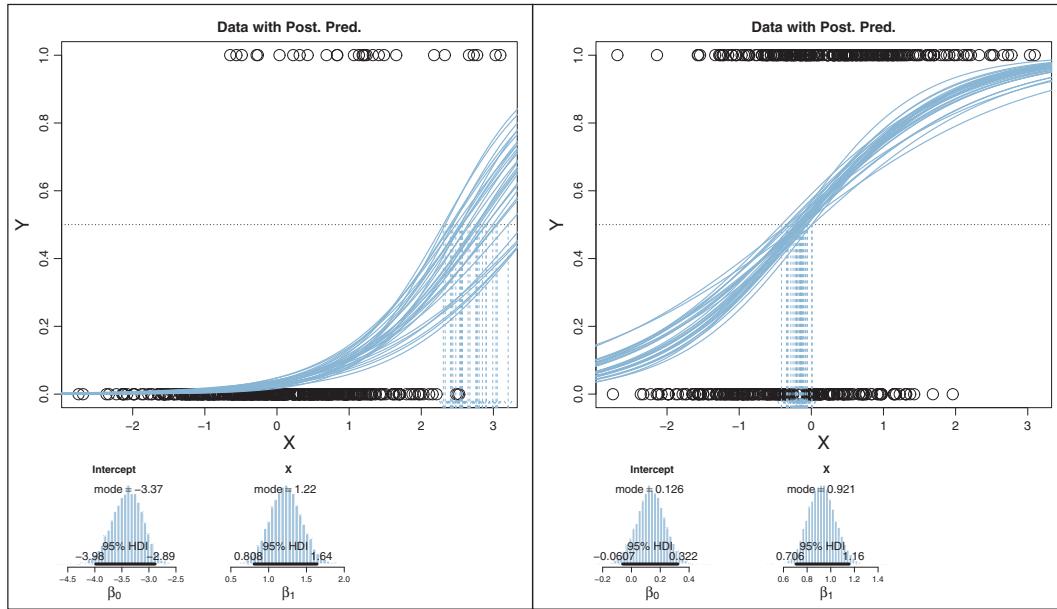


Figure 21.5 Parameter estimates are more uncertain when there are few 0's or 1's in the data. The main panels have data with the same x values, and y values randomly generated by logistic functions with the same slope ($\beta_1 = 1$) but different intercepts ($\beta_0 = -3$ and $\beta_0 = 0$).

any independent variables, then you should be aware that the parameter estimates may be surprisingly ambiguous if your data have only a small proportion of 0's or 1's. Conversely, when interpreting the parameter estimates, it helps to recognize when the threshold falls at the fringe of the data, which is one possible cause of relatively few 0's or 1's.

21.2.3. Correlated predictors

Another important cause of parameter uncertainty is correlated predictors. This issue was previously discussed at length, but the context of logistic regression provides novel illustration in terms of level contours.

Figure 21.6 shows a case of data with strongly correlated predictors. Both predictors have values randomly drawn from a standardized normal distribution. The y values are generated randomly from a Bernoulli distribution with bias given by slope parameters $\beta_1 = 1$ and $\beta_2 = 1$. The threshold falls in their midst, with $\beta_0 = 0$, so there are about half 0's and half 1's.

The posterior estimates of the parameter values are shown in Figure 21.6. In particular, notice that the 50% level contours (the threshold lines) are extremely ambiguous, with many different possible angles. The right panel shows strong anticorrelation of credible β_1 and β_2 values.

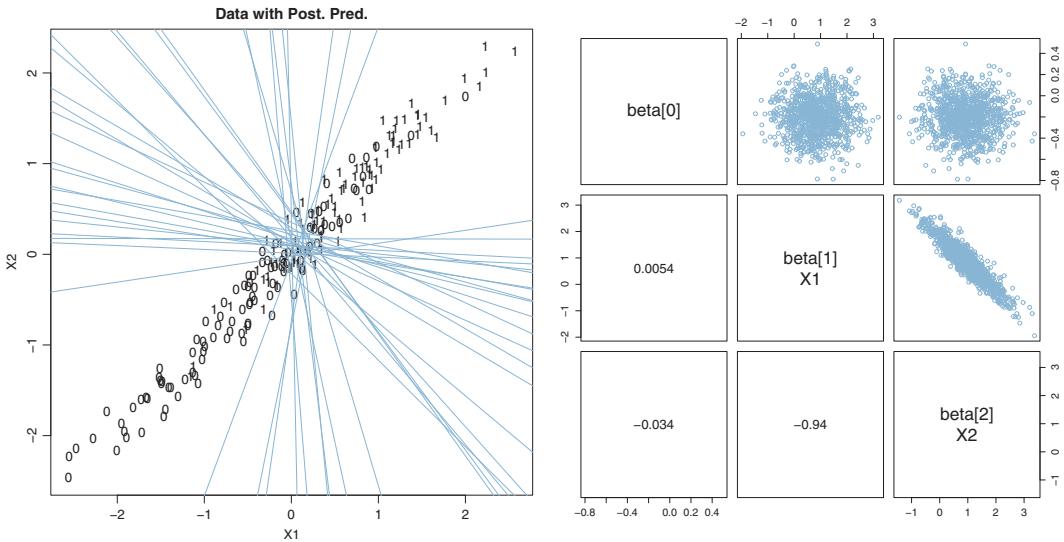


Figure 21.6 Estimates of slope parameters trade off when the predictors are correlated. Left panel shows credible 50% level contours superimposed on data. Right panel shows strong anticorrelation of credible β_1 and β_2 values.

The same ambiguity can arise for correlations of multiple predictors, but higher-dimensional correlations are difficult to graph. As in the case of linear regression, it is important to consider the correlations of the predictors when interpreting the parameters of logistic regression. The high-level scripts display the predictor correlations on the R console.

21.2.4. Interaction of metric predictors

There may be applications in which it is meaningful to consider a multiplicative interaction of metric predictors. For example, in the case of predicting gender (male vs female) from weight and height, it could be that an additive combination of the predictors is not very accurate. For example, it might be that only the combination of tall and heavy is a good indicator of being male, while being tall but not heavy, or heavy but not tall, both indicate being female. This sort of conjunctive combination of predictors can be expressed by their multiplication (or other ways).

Figure 21.7 shows examples of logistic surfaces with multiplicative interaction of predictors. Within Figure 21.7, the title of each panel displays the formula being plotted. The left column of Figure 21.7 shows examples without interaction, for reference. The right column shows the same examples from the left column but with a

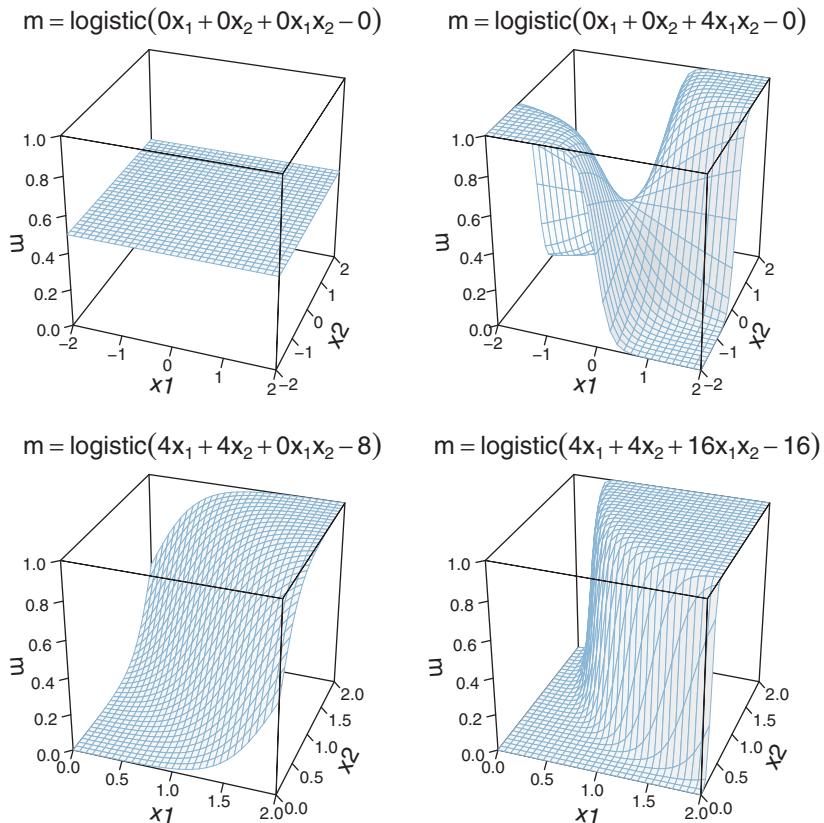


Figure 21.7 Multiplicative interaction of metric predictors in logistic regression. Left column shows examples of no interaction. Right column shows corresponding logistic surfaces with interaction. Title of each plot shows the coefficients on the predictors.

multiplicative interaction included. In the top row, you can see that including the term $+4x_1x_2$ makes the logistic surface rise whenever x_1 and x_2 are both positive or both negative. In the bottom row, you can see that including a multiplicative interaction (with an adjustment of the intercept) produces the sort of conjunctive combination described in the previous paragraph. Thus, when either x_1 or x_2 is zero, the logistic surface is nearly zero, but when both predictors are positive, the logistic surface is high. Importantly, the 50% level contour is straight when there is no interaction, but is curved when there is interaction.

A key thing to remember when using an interaction term is that regression coefficients on individual predictors only indicate the slope when all the other predictors are set to zero. This issue was illustrated extensively in the context of linear regression in Section 18.2 (p. 525). Analogous points apply to logistic regression.

21.3. ROBUST LOGISTIC REGRESSION

Look again at [Figure 21.3](#) (p. 627), which showed gender as a function of weight alone. Notice that there are several unusual data points in the lower right that represent heavy females. For these data points to be accommodated by a logistic function, its slope must not be too extreme. If the slope is extreme, then the logistic function gets close to its asymptote at $y = 1$ for heavy weights, which then makes the probability of data points with $y = 0$ essentially nil. In other words, the outliers relative to the logistic function can only be accommodated by reducing the magnitude of the estimated slope.

One way to address the problem of outliers is by considering additional predictors, such as height in addition to weight. It could be that the heavy females also happen to be short, and therefore a logistic function with a large slope coefficient on height would account for the data, without having to artificially reduce its slope coefficient on weight.

Usually we do not have other predictors immediately at hand to account for the outliers, and instead we use a model that incorporates a description of outliers as such. Because this sort of model provides parameter estimates that are relatively stable in the presence of outliers, it is called robust against outliers. We have routinely considered robust models in previous chapters involving metric variables (e.g., Section 16.2, p. 458). In the present context of a dichotomous predicted variable, we need a new mathematical formulation.

We will describe the data as being a mixture of two different sources. One source is the logistic function of the predictor(s). The other source is sheer randomness or “guessing,” whereby the y value comes from the flip of a fair coin: $y \sim \text{Bernoulli}(\mu = 1/2)$. We suppose that every data point has a small chance, α , of being generated by the guessing process, but usually, with probability $1 - \alpha$, the y value comes from the logistic function of the predictor. With the two sources combined, the predicted probability that $y = 1$ is

$$\mu = \alpha \cdot \frac{1}{2} + (1 - \alpha) \cdot \text{logistic} \left(\beta_0 + \sum_j \beta_j x_j \right) \quad (21.2)$$

Notice that when the guessing coefficient is zero, then the conventional logistic model is completely recovered. When the guessing coefficient is one, then the y values are completely random. (This could also be achieved by setting all the slope coefficients in the logistic function to zero.)

Our goal is to estimate the guessing coefficient along with the logistic parameters. For this, we need to establish a prior on the guessing coefficient, α . In most applications we would expect the proportion of random outliers in the data to be small, and therefore the prior should emphasize small values of α . For the example presented below, I have set the prior as $\alpha \sim \text{dbeta}(1, 9)$. This prior distribution gives values of α greater than 0.5 very low but non-zero probability.

The JAGS model specification for robust logistic regression is a modest extension of the model specification for ordinary logistic regression. As before, it starts with standardizing the data and ends with transforming the parameters back to the original scale. The mixture of [Equation 21.2](#) is directly expressed in JAGS, with α coded as `guess`:

```
for ( i in 1:Ntotal ) {
  y[i] ~ dbern( mu[i] )
  mu[i] <- ( guess * (1/2)
    + (1.0-guess) * ilogit(zbeta0+sum(zbeta[1:Nx] * zx[i,1:Nx])) )
}
guess ~ dbeta(1,9)
```

The model definitions are in the file named `Jags-Ydich-XmetMulti-Mlogistic.R` and the high-level script that calls the model is named `Jags-Ydich-XmetMulti-MlogisticRobust-Example.R`.

[Figure 21.8](#) shows the fit of the robust logistic regression model for predicting gender as a function of weight alone. The superimposed curves show μ from [Equation 21.2](#). Notice that the curves asymptote at levels away from 0 and 1, unlike the ordinary logistic curves in [Figure 21.3](#). The modal estimate of the guessing parameter is almost 0.2. This implies that the asymptotes are around $y = 0.1$ and $y = 0.9$. Especially for heavy weights, the non-1 asymptote lets the model accommodate outlying data while still having a steep slope at the threshold. Notice that the slope of the logistic is larger than in [Figure 21.3](#).

[Figure 21.9](#) shows pairwise posterior parameters. In particular, consider the panel that plots the value of β_1 (i.e., the slope on weight) against the value of the guessing parameter. Notice the strong positive correlation between those two parameters. This means that as the guessing parameter gets larger, credible values of the slope get larger also.

For some data sets with extreme outliers (relative to the predictions of ordinary logistic regression), the inclusion of a guessing parameter for robust logistic regression can make the difference between slope estimates that are unrealistically small and slope estimates that are large and certain enough to be useful. But if the data do not have extreme outliers, they can be difficult to detect, as exemplified in [Exercise 21.1](#). There are other ways to model outliers, as we will see in Exercise 23.2 (p. 701) in the chapter on ordinal predicted variables.

21.4. NOMINAL PREDICTORS

We now turn our attention from metric predictors to nominal predictors. For example, we might be interested in predicting whether a person voted for candidate A or candidate B (the dichotomous predicted variable), based on the person's political party and religious affiliations (nominal predictors). Or we might want to predict the probability that a

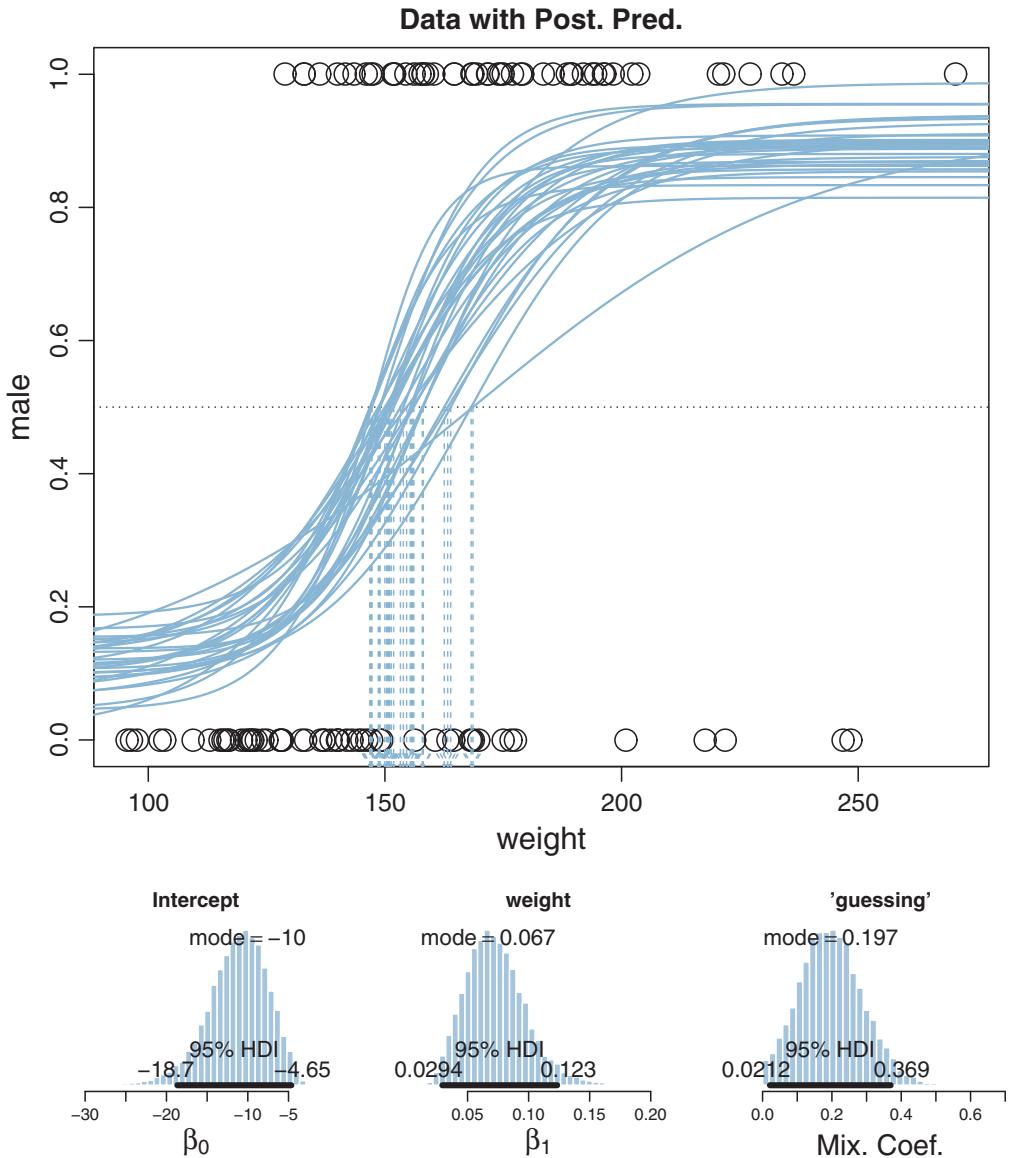


Figure 21.8 Predicting gender (arbitrarily coded as male = 1, female = 0) as a function of weight (in pounds), using robust logistic regression. Upper panel: Data are indicated by dots, the same as in [Figure 21.3](#). Curves are a random sample from the MCMC posterior; notice asymptotes away from 0,1 limits. Descending arrows point to threshold weights at which the probability of male is 50%. Lower panels: Marginal posterior distribution on baseline, slope, and guessing coefficient. Pairwise plots are shown in [Figure 21.9](#).

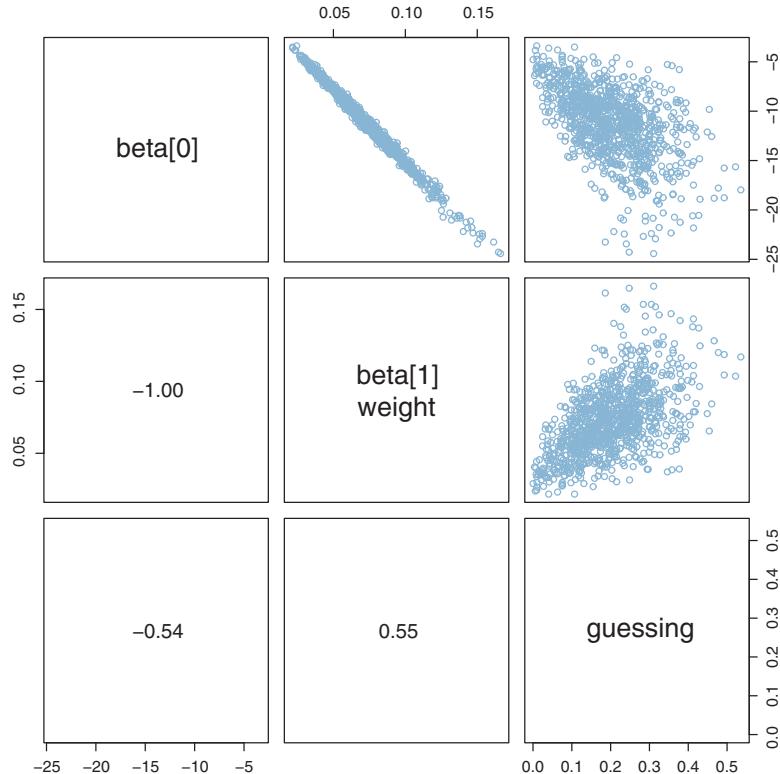


Figure 21.9 Pairwise plots for posterior distribution in Figure 21.8. Notice here that the guessing coefficient is correlated with the slope, beta[1].

baseball player will get a hit when at bat (dichotomous predicted), based on the player's fielding position (nominal predictor).

21.4.1. Single group

We start with the simplest, trivial case: A single group. In this case, we have several observed 0's and 1's, and our goal is to estimate the underlying probability of 1's. This is simply the case of estimating the underlying bias of a coin by observing several flips. The novelty here is that we treat it as a case of the generalized linear model, using a logistic function of a baseline.

Back in Chapter 6, we formalized this situation by denoting the bias of the coin as θ , and giving θ a beta prior. Thus, the model was $y \sim \text{Bernoulli}(\theta)$ with $\theta \sim \text{dbeta}(a, b)$. Now, the bias of the coin is denoted μ (instead of θ). The value of μ is given by a logistic function of a baseline, β_0 . Thus, in the new model:

$$y \sim \text{Bernoulli}(\mu) \quad \text{with} \quad \mu = \text{logistic}(\beta_0)$$

The model expresses the bias in the coin in terms of β_0 . When $\beta_0 = 0$, the coin is fair because $\mu = 0.5$. When $\beta_0 > 0$, then $\mu > 0.5$ and the coin is head biased. When $\beta_0 < 0$, then $\mu < 0.5$ and the coin is tail biased. Notice that β_0 ranges from $-\infty$ to $+\infty$, while μ ranges from 0 to 1. Therefore the prior on β_0 must also support $-\infty$ to $+\infty$. The conventional prior is a normal distribution:

$$\beta_0 \sim \text{normal}(M_0, S_0)$$

When $M_0 = 0$, the prior distribution is centered at $\mu = 0.5$. [Figure 21.10](#) presents a diagram of the model, which should be compared with the beta prior in Figure 8.2 (p. 196).

Because μ is a logistic function of a normally distributed value β_0 , it is not immediately obvious what the prior distribution of μ actually looks like. [Figure 21.11](#) gives some examples. In each example, random values of β_0 were generated from its normal prior, and then those values of β_0 were converted to values of μ via the logistic function. The resulting values of μ were then plotted in a histogram. The histograms in [Figure 21.11](#) are outlined by the exact mathematical form of the implied prior on μ , derived from the reparameterization formula explained in Section 25.3 (p. 729).²

For some values of prior mean and standard deviation on β_0 , the prior distribution of μ looks somewhat similar to a beta distribution. But there are many choices of prior mean and standard deviation that make the distribution of μ look nothing like a beta distribution. In particular, there is no choice of prior mean and standard deviation that give μ a uniform distribution. Also, while the beta-prior constants have a natural interpretation in terms of previously observed data (recall Section 6.2.1, p. 127), the normal-prior constants for the logistic mapping to μ are not as intuitive.

It is straight forward to convert the beta-prior program, `Jags-Ydich-Xnom1subj-MbernBeta.R`, to a version that uses the logistic of a baseline. (The new program could be called `Jags-Ydich-Xnom1subj-Mlogistic.R`.) The model specification from `Jags-Ydich-Xnom1subj-MbernBeta.R` is as follows:

```
model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dbern( theta )
  }
  theta ~ dbeta(1,1)
}
```

² The reparameterization formula, Equation 25.1 (p. 729) in Section 25.3, requires the inverse and derivative of the logistic function. We already know that the inverse of the logistic is the logit, which is $\text{logistic}^{-1}(\mu) = \log(\mu/(1 - \mu))$. It turns out that the derivative of the logistic function is conveniently expressed in terms of the logistic function itself: $\text{logistic}'(\beta) = \text{logistic}(\beta) \cdot (1 - \text{logistic}(\beta))$. We substitute $\text{logistic}^{-1}(\mu)$ for β in the derivative. Then Equation 25.1 (p. 729) is used to plot the probability density of μ .

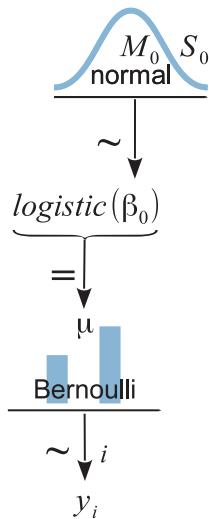


Figure 21.10 Estimating the underlying bias of a coin using a logistic function of a baseline. When $\beta_0 = 0$, then $\mu = 0.5$, and the coin is fair. Compare with the beta prior in Figure 8.2 (p. 196).

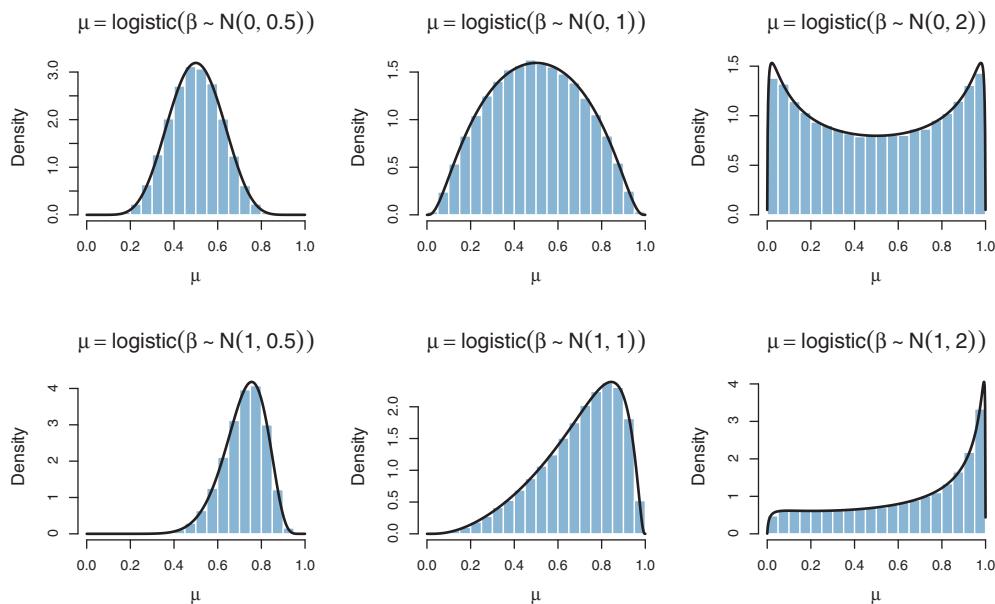


Figure 21.11 Prior distributions on μ for different choices of M_0 and S_0 in Figure 21.10.

The logistic version could instead use the following:

```
model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dbern( ilogit( b0 ) )
  }
  b0 ~ dnorm( 0 , 1/2^2 )
}
```

The prior on β_0 was given a mean of $M_0 = 0$ and a standard deviation of $S_0 = 2$. Notice that the model specification, above, did not use a separate line to explicitly name μ . If you want to keep track of μ , you could do it after the fact by creating $\mu = \text{logistic}(\beta_0)$ at each step in the chain, or you could do it this way:

```
model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dbern( mu )
  }
  mu <- ilogit( b0 )
  b0 ~ dnorm( 0 , 1/2^2 )
}
```

Review Section 17.5.2 (p. 502) for tips about what other aspects of the program would need to be changed. For any given data set, the posterior distribution created by the logistic version of the program would not exactly match the posterior distribution created by the beta version of the program, because the prior distributions are not identical. You have the chance to try all this, hands on, in [Exercise 21.2](#).

21.4.2. Multiple groups

The previous section discussed the trivial case of a single group, primarily for the purpose of introducing concepts. In the present section, we consider a more realistic case in which there are several groups. Moreover, instead of each “subject” within a group contributing a single dichotomous score, each subject contributes many dichotomous measures.

21.4.2.1 Example: Baseball again

We will focus on the baseball data introduced in Section 9.5.1 (p. 253), in which each baseball player has some opportunities at bat and gets a hit on some of those occasions. Thus, each opportunity at bat has a dichotomous outcome. We are interested in estimating the probability of getting a hit based on the primary fielding position of the player. Position is a nominal predictor.

In this situation, each player contributes many dichotomous outcomes, not just one. Moreover, each player within a position is thought to have his own batting ability distinct from other players, although all players are thought to have abilities that are representative

of their position. Thus, each position is described as having an underlying ability, and each player is a random variant of that position's ability, and the data from a player are generated by the individual player's ability.

21.4.2.2 The model

The model we will use is diagrammed in Figure 21.12. The top part of the structure is based on the ANOVA-like model of Figure 19.2 (p. 558). It shows that the modal ability of position j , denoted ω_j , is a logistic function of a baseline ability β_0 plus a deflection for the position, $\beta_{[j]}$. The baseline and deflection parameters are given the usual priors for an ANOVA-style model.

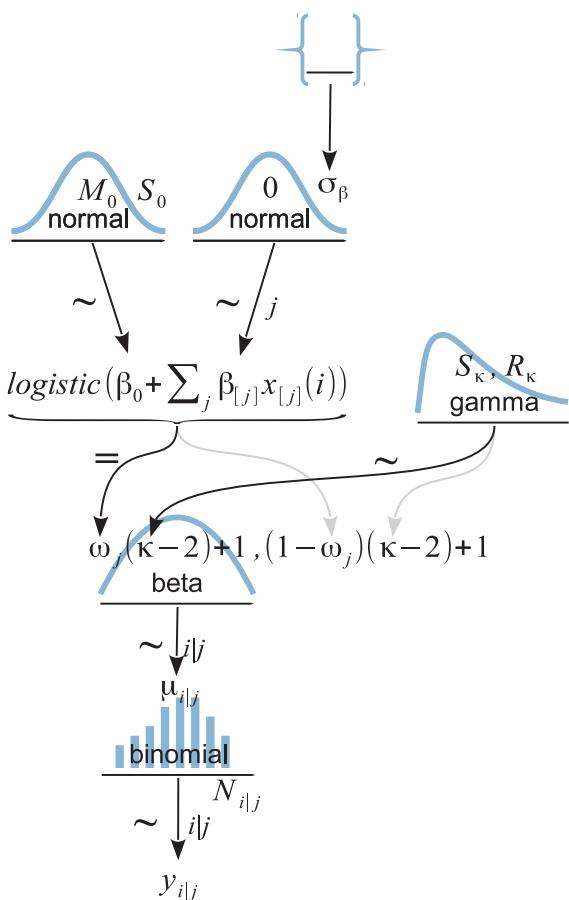


Figure 21.12 Hierarchical diagram for logistic ANOVA-like model. The top part of the structure is based on the ANOVA-like model of Figure 19.2 (p. 558). The lower part of the structure is based on the models of Figure 9.7 (p. 236) and Figure 9.13 (p. 252).

The lower part of [Figure 21.12](#) is based on the models of Figure 9.7 (p. 236) and Figure 9.13 (p. 252). The ability of position j , denoted ω_j , is the mode of a beta distribution that describes the spread of individual abilities within the position. The concentration of the beta distribution, denoted κ , is estimated and given a gamma prior. I've chosen to use a single concentration parameter to describe all groups simultaneously, by analogy to an assumption of homogeneity of variance in ANOVA. (Group-specific concentration parameters could be used, as shown in Figure 9.13, p. 252.) The ability of player i in position j is denoted μ_{ij} . The number of hits by player i , out of N_{ij} opportunities at bat, is denoted y_{ij} , and is assumed to be a random draw from a binomial distribution with mean μ_{ij} . We use the binomial distribution as a computational shortcut for multiple independent dichotomous events, not as a claim that the measuring event was conditional on fixed N , as was discussed in Section 9.4 (p. 249; see also Footnote 5, p. 249).

The model functions are in the file name `Jags-Ybinom-Xnom1fac-Mlogistic.R`, and the high-level script is the file named `Jags-Ybinom-Xnom1fac-Mlogistic-Example.R`. The JAGS model specification implements all the arrows in [Figure 21.12](#). I hope that by this point in the book you can scan the JAGS model specification, below, and understand how each line of code corresponds with the arrows in [Figure 21.12](#):

```
model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dbin( mu[i] , N[i] )
    mu[i] ~ dbeta( omega[x[i]]*(kappa-2)+1, (1-omega[x[i]])*(kappa-2)+1 )
  }
  for ( j in 1:NxLvl ) {
    omega[j] <- ilogit( a0 + a[j] ) # In JAGS, ilogit is logistic
    a[j] ~ dnorm( 0.0 , 1/aSigma^2 )
  }
  a0 ~ dnorm( 0.0 , 1/2^2 )
  aSigma ~ dgamma( 1.64 , 0.32 ) # mode=2, sd=4
  kappa <- kappaMinusTwo + 2
  kappaMinusTwo ~ dgamma( 0.01 , 0.01 ) # mean=1, sd=10 (generic vague)
  # Convert a0,a[] to sum-to-zero b0,b[] :
  for ( j in 1:NxLvl ) { m[j] <- a0 + a[j] } # cell means
  b0 <- mean( m[1:NxLvl] )
  for ( j in 1:NxLvl ) { b[j] <- m[j] - b0 }
}
}
```

The only novelty in the code, above, is the choice of constants in the prior for `aSigma`. Recall that `aSigma` represents σ_β in [Figure 21.12](#), which is the standard deviation of the deflection parameters across groups. The gamma prior on `aSigma` was made broad on its scale. The reasonable values of σ_β are not very intuitive because they refer to standard deviations of deflections that are the inverse-logistic of probabilities. Consider two extreme probabilities near 0.001 and 0.999. The corresponding inverse-logistic

values are about -6.9 and $+6.9$, which have a standard deviation of almost 10. This represents the high end of the prior distribution on σ_β , and more typical values will be around, say, 2. Therefore the gamma prior was given a mode of 2 and a standard deviation of 4. This is broad enough to have minimal influence on moderate amounts of data. You should change the prior constants if appropriate for your application.

21.4.2.3 Results

Each player's ratio of hits to at-bats is shown as a dot in Figure 21.13, with the dot size indicating the number of at-bats. Larger dots correspond to more data, and therefore should be more influential to the parameter estimation. The posterior predictive distribution is superimposed along the side of each group's data. The profiles are beta distributions from the model of Figure 21.12 with credible values of ω_j and κ . The beta distributions have their tails clipped so they span 95%. Each "moustache" is very thin, meaning that the estimate of the group modes is fairly certain. This high degree of certainty makes intuitive sense, because the data set is fairly large. The model assumed equal concentration in every group, which might be a poor description of the data. In particular, visual inspection of Figure 21.13 suggests that there might be greater variation within pitchers than within other positions.

Contrasts between positions are shown in the lower panels of Figure 21.13. The contrasts are shown for both the log odds deflection parameters (β_j or b) and the group modes (ω or omega). Compare the group-mode contrasts with the previously reported contrasts shown in Figure 9.14 (p. 256). They are very similar, despite the difference in model structure.

The present model focuses on descriptions at the group level, not at the individual-player level. Therefore the program does not have contrasts for individual players built in. But the JAGS model does record the individual-player estimates as $\text{mu}[i]$, so contrasts of individual players can be conducted.

Which model is better, the ANOVA-style model of Figure 21.12 in this section, or the tower of beta distributions used in Figures 9.7 (p. 236) and 9.13 (p. 252)? The answer is: It depends. It depends on meaningfulness, descriptive adequacy, and generalizability. The tower of beta distributions might be more intuitively meaningful because the mode parameters of the beta distributions can be directly interpreted on the scale of the data, unlike the deflection parameters inside the logistic function. Descriptive adequacy for these data might demand that each group has its own concentration parameter, regardless of upper-level structure of the model. The ANOVA-style model may be more generalizable for applications that have multiple predictors, including covariates. Indeed, the top structure of Figure 21.12 can be changed to any combination of metric and nominal predictors that are of interest.

Data with Posterior Predictive Distrib.

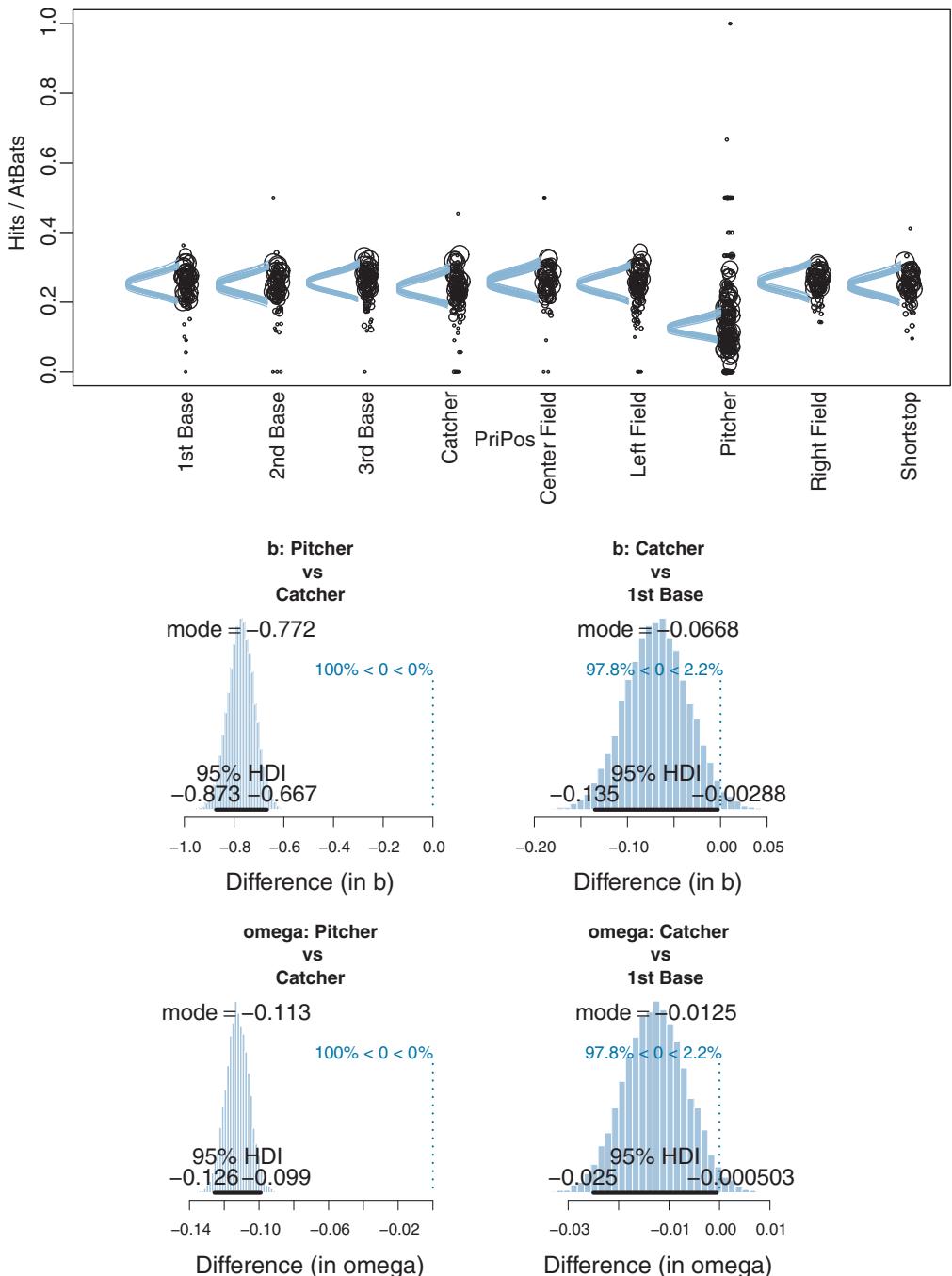


Figure 21.13 Baseball batting data are shown in the upper panel with dot size proportional to number of at-bats. The posterior predictive distributions are credible beta distributions assuming homogeneous concentration across positions. Lower panels show selected contrasts, which can be compared with the contrasts shown in Figure 9.14 (p. 256).

21.5. EXERCISES

Look for more exercises at <https://sites.google.com/site/doingbayesiandataanalysis/>

Exercise 21.1. [Purpose: Robust logistic regression: Building understanding and seeing its usefulness and difficulty.] When there are extreme outliers, robust logistic regression makes a big difference in the parameter estimates. But when the outliers are only modest in extremity or numerosity, they can be difficult to detect.

(A) Create some data with outliers. Type the following into R:

```
N=500                      # number of data points to generate.
x = runif(N)                # random values from a uniform distribution.
x = (x-mean(x))/sd(x)      # standardize the x values.
b0 = 0                      # the true intercept.
b1 = 4                      # the true slope.
guess = 0.1                 # the true guessing amount.
# Probability of y=1 is mixture of guessing and logistic:
mu = guess*(1/2) + (1-guess)*1/(1+exp(-(b0+b1*x)))
# Generate random y values:
y = rep(NA,N)   # set up place holder for y values
for ( i in 1:length(x) ) {
  y[i] = sample( c(1,0) , size=1 , prob=c(mu[i],1-mu[i]) )
}
# Write the data into a file:
write.csv( cbind( Y=y , X=x ) ,
           file=paste0("RobustLogisticExercise-",b0,"-",b1,"-", guess,".csv") ,
           row.names=FALSE )
```

Make sure that the current working directory is set appropriately so that the saved data file is where you want it.

(B) Run the nonrobust logistic regression on the data. In the top part of the high-level script, Jags-Ydich-XmetMulti-Mlogistic-Example.R, use the following to read in the data file:

```
myData = read.csv("RobustLogisticExercise-0-4-0.1.csv")
yName = "Y" ; xName = "X"
fileNameRoot = "RobustLogisticExercise-0-4-0.1-REGULAR-"
numSavedSteps=15000 ; thinSteps=2
```

In your report, include the graph of the data with superimposed logistic curves, and the marginal posterior distribution on β_1 . Is β_1 accurately estimated?

(C) Run the robust logistic regression on the data. In the top part of the high-level script, Jags-Ydich-XmetMulti-MlogisticRobust-Example.R, use the following to read in the data file:

```
myData = read.csv("RobustLogisticExercise-0-4-0.1.csv")
yName = "Y" ; xName = "X"
fileNameRoot = "RobustLogisticExercise-0-4-0.1-ROBUST-"
numSavedSteps=15000 ; thinSteps=2
```

In your report, include the graph of the data with superimposed logistic curves, and the marginal posterior distribution on β_1 . Is β_1 accurately estimated? Is the estimate of β_1 different than for the nonrobust model of the previous part?

(D) Repeat the above three parts, but this time generate random x values from a normal distribution instead of from a uniform distribution. That is, change a single line, from $x = runif(N)$ to $x = rnorm(N)$. You should find that the robust logistic regression does not recover the true slope (or guessing rate) very well, although a little better than nonrobust logistic regression. Discuss why. The moral of the examples was stated at the beginning of the exercises. Is there any disadvantage to using robust logistic regression?

Exercise 21.2. [Purpose: Practice modifying JAGS programs, and comparing the beta prior with the logistic-normal prior for the bias of a coin.]

(A) Create a program for estimating the bias of a single coin, using a logistic-normal prior, as outlined in [Section 21.4.1](#) (p. 638). Hint: Make a copy of `Jags-Ydich-Xnom1subj-MbernBeta.R` and call it `Jags-Ydich-Xnom1subj-Mlogistic.R`. Make the required changes in the copied program. Do the same for the high-level script, `Jags-Ydich-Xnom1subj-MbernBeta-Example.R`.

(B) Create graphs of the prior on μ (review Section 8.5, p. 211, for how). See [Figure 21.11](#) for reference.

(C) Consider a data set with 1 head in 1 flip. Show the posterior distribution for the new model, and for the beta-prior model. Are the posterior distributions noticeably different?

(D) Consider a data set with 30 heads in 40 flips. Show the posterior distribution for the new model, and for the beta-prior model. And the posterior distributions noticeably different?

Exercise 21.3. [Purpose: Agony and frustration for little gain while extending the logistic “ANOVA” model to have different concentration parameters for different groups. Sort of like climbing Everest, but it’s less fun to tell people about this afterwards.] The baseball data in [Figure 21.13](#) suggested that some positions (especially pitchers) might have different ranges of individual abilities than other positions. In other words, the model’s assumption of equal concentration (k) in all groups might be inadequate. For this exercise, your task is to extend the model of [Figure 21.12](#)

so it has distinct concentration parameters for each group. Use Figure 9.13 (p. 252) as a guide for distinct κ_j without estimating their variability. Or, for overachievers, use Figure 19.6 (p. 574) as a guide for distinct κ_j *with* estimating their variability. Copy the files Jags-Ybinom-Xnom1fac-Mlogistic.R and Jags-Ybinom-Xnom1fac-Mlogistic-Example.R to new files, renamed appropriately, for your starting point.