Exercises week 2

prof. Richard Torkar

March 28, 2021

You cannot answer a question that you cannot ask, and you cannot ask a question that you have no words for.

Judea Pearl

The weekly exercises are mainly from McElreath's notes which can be found on GitHub. `https://github.com/rmcelreath/stat_rethinking_2020/tree/main/homework`

Exercise 1. The weights listed below were recorded in the !Kung census, but heights were not recorded for these individuals. Provide predicted heights and 89% compatibility intervals for each of these individuals. That is, fill in the table below, using model-based predictions.

individual	weight	expected height	89% interval
1	46.95		
2	43.72		
3	64.78		
4	32.59		
5	54.63		

Exercise 2. Select out all the rows in the `Howell1` data with ages below 18 years of age. If you do it right, you should end up with a new data frame with 192 rows in it.

(a) Fit a linear regression to these data, using `quap`. Present and interpret the estimates. For every 10 units of increase in weight, how much taller does the model predict a child gets?

(b) Plot the raw data, with height on the vertical axis and weight on the horizontal axis. Superimpose the MAP regression line and 89% interval for the mean. Also superimpose the 89% interval for predicted heights.

(c) What aspects of the model fit concern you? Describe the kinds of assumptions you would change, if any, to improve the model. You don't have to write any new code. Just explain what the model appears to be doing a bad job of, and what you hypothesize would be a better model.

Solution 1. First, fit the model with `quap`, to provide a quadratic approximation of the posterior. This is just as in the chapter.

```
library(rethinking)
data(Howell1)
d <- Howell1
d2 <- d[ d$age >= 18 , ]
xbar <- mean(d2$weight)
m <- quap(
    alist(
        height ~ dnorm( mu , sigma ) ,
        mu <- a + b*( weight - xbar ) ,
        a ~ dnorm( 178 , 20 ) ,
        b ~ dlnorm( 0 , 1 ) ,
        sigma ~ dunif( 0 , 50 )
) , data=d2 )
```

Then produce samples from the quadratic approximate posterior. You could use `mvrnorm` directly, or the convenient `extract.samples` function:

```
post <- extract.samples( m )
str(post)
```

which gives the following output,

```
'data.frame': 10000 obs. of 3 variables:
$ a :num 155 155 155 155 155...
$ b : num 0.992 0.903 0.92 0.835 0.971 ...
$ sigma: num 5.25 4.98 4.86 5.23 5.01 ...
- attr(*, "source")= chr "quap posterior: 10000 samples from m"
```

Now we want to plug the weights in the table into this model, and then average over the posterior to compute predictions for each individual's height. The question is ambiguous as to whether it wants μ only or rather the distribution of individual height measurements (using σ). We'll show the harder approach, that uses σ and simulates individual heights. Also, we won't use `link` or `sim`, but it's fine if you did.

For the first individual, the code might look like this:

```
y <- rnorm( 1e5 , post$a + post$b*( 46.95 - xbar ) , post$sigma )
mean(y)
PI(y,prob=0.89)
```

giving us something like this,

```
[1] 156.364
      5%      94%
148.2133 164.4904
```

How does the code work? The first line, which includes `rnorm`, simulates 100-thousand heights, using the samples from the posterior and an assumed weight of 46.95 kg. The second line then computes the average of these simulated heights. That gives the expected (mean) height. The third line then computes the 89% compatibility interval of height.

To do the above for each row in the table, you can just replace the weight each time. We're going to write a function and use the R function sapply:

```
f <- function( weight ) {
    y <- rnorm( 1e5 , post$a + post$b*( weight - xbar ) , post$sigma )
    return( c( mean(y) , PI(y,prob=0.89) ) )
}
weight_list <- c(46.95,43.72,64.78,32.59,54.63)
result <- sapply( weight_list , f )
```

Now let's format into a results table:

```
rtab <- cbind( weight_list , t( result ) )
colnames(rtab) <- c("weight","height","5%","94%")
rtab
```

which provides this,

```
     weight   height       5%      94%
[1,]  46.95 156.3754 148.2310 164.5441
[2,]  43.72 153.4612 145.3171 161.6234
[3,]  64.78 172.4761 164.1917 180.7644
[4,]  32.59 143.3843 135.2444 151.5570
[5,]  54.63 163.2732 155.1527 171.4127
```

You could have also computed the expected heights straight from the MAP. That approach is fine, and will give nearly the same answer.

Solution 2.

(a) First make a new data frame with just the non-adults in it:

```
library(rethinking)
data(Howell1)
d <- Howell1
d3 <- d[ d$age < 18 , ]
str(d3)
```

which gives something like this,

```

'data.frame': 192 obs. of 4 variables:
$ height: num 121.9 105.4 86.4 129.5 109.2 ...
$ weight: num 19.6 13.9 10.5 23.6 16 ...
$age :num 12 8 6.5 13 7 17 16 11 17 8...
$male :int 1 0 0 1 0 1 0 1 0 1...
```

Now find the quadratic approximate posterior. The code is the same as before. The data frame just changes.

```
xbar <- mean( d3$weight )
m <- quap(
    alist(
        height ~ dnorm( mu , sigma ) ,
        mu <- a + b*( weight - xbar ) ,
        a ~ dnorm( 178 , 20 ),
        b ~ dnorm( 0 , 10 ) ,
        sigma ~ dunif( 0 , 50 )
    ) , data=d3 )
precis(m)
```

gives this,

```
         mean   sd   5.5%  94.5%
a       108.38 0.61 107.41 109.36
b       2.72   0.07   2.61   2.83
sigma   8.44   0.43   7.75   9.13
```

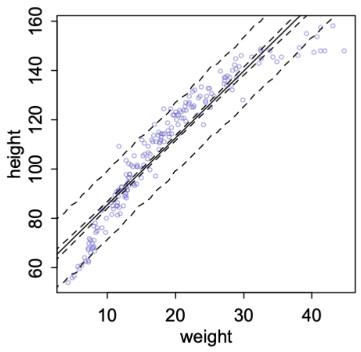
The estimates suggest that the MAP coefficient for weight is 2.7. This implies that for a unit change of 1kg of weight, we predict an average of 2.7cm of increase in height.

(b) Now to plot the raw data and superimpose the model estimates, modify the code in Chapter 4. We will sample from the naive posterior, then compute 89% intervals for the mean and predicted heights. This is what the complete code looks like, if you opt not to use the convenience functions `link` and `sim`:

```
post <- extract.samples( m )
w.seq <- seq(from=1,to=45,length.out=50)
mu <- sapply( w.seq , function(z) mean( post$a + post$b*(z-xbar) ) )
mu.ci <- sapply( w.seq , function(z)
    PI( post$a + post$b*(z-xbar) , prob=0.89 ) )
pred.ci <- sapply( w.seq , function(z)
    PI( rnorm( 10000 , post$a + post$b*(z-xbar) , post$sigma) , 0.89 ) )
```

And then to plot everything:

```
plot( height ~ weight , data=d3 ,
    col=col.alpha("slateblue",0.5) , cex=0.5 )
lines( w.seq , mu )
lines( w.seq , mu.ci[1,] , lty=2 )
lines( w.seq , mu.ci[2,] , lty=2 )
lines( w.seq , pred.ci[1,] , lty=2 )
lines( w.seq , pred.ci[2,] , lty=2 )
```



And the resulting plot looks like the one to the right.

(c) The major problem with this model appears to be that the relationship between weight and height, for non-adults, isn't very linear. Instead it is curved (see plot to the right). As a result, at low weight values, the predicted mean is above most of the actual heights. At middle weight values, the predicted mean is below most of the heights. Then again at high weight values, the mean is above the heights.

A parabolic model would likely fit these data much better. But that's not the only option. What we're after essentially is some way to model a reduction of the slope between height and weight, as weight increases.