Chapter 6 Questions - Bayesian Statistics

Melissa Van Bussel

June 25, 2018

Chapter 6 points: 2 + 9 + 4 = 15 points total.

Easy (2 points total)

6E1

The three criteria are: continuity, increase with number of events, and additivity.

6E2

```
p <- c(0.7, 0.3)
-1 * sum(p * log(p))
```

```
## [1] 0.6108643
```

6E3

```
p <- c(0.2, 0.25, 0.25, 0.3)
-1 * sum(p * log(p))
```

```
## [1] 1.376227
```

6E4

```
p <- c(1/3, 1/3, 1/3)
-1 * sum(p * log(p))
```

```
## [1] 1.098612
```

Medium (9 points total)

6M2 (2 points)

Model selection is picking a model based on diagnostics (such as information criteria). Model averaging is weighting each model by its distance from the "best model".

6M3 (2 points)

It wouldn't make sense to compare 2 models that are on a different number of observations because that means you're comparing models on different data. One model might appear better than the other because the data is better, which would have nothing to do with the model itself.

6M4 (2 points)

When a prior becomes more concentrated/specific in one area, it means that there is less freedom for the other parameters to move around. This means that the model will be less flexible, and the DIC and WAIC will reflect this.

6M5 (1 point)

Informative priors reduce overfitting of a model by reducing the sensitivity of a model to a sample.

6M6 (2 points)

If you give a prior that is TOO informative, the model will not be able to "learn" properly, so the best values will not be found and the result will be underfitting.

Hard (4 points total)

```r
library(rethinking)
```

```
## Loading required package: rstan
```

```
## Warning: package 'rstan' was built under R version 3.3.3
```

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

```
## Warning: package 'ggplot2' was built under R version 3.3.3
```

```
## Loading required package: StanHeaders
```

```
## Warning: package 'StanHeaders' was built under R version 3.3.3
```

```
## rstan (Version 2.17.3, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
## Loading required package: parallel
```

```
## rethinking (Version 1.59)
```

```r
data(Howell1)
d <- Howell1
d$age <- (d$age - mean(d$age)) / sd(d$age)
set.seed(1000)
i <- sample(1:nrow(d), size = nrow(d) / 2)
d1 <- d[i, ]
d2 <- d[-i, ]
```

Next, we fit all of the models:

```
model1 <- map(alist(
  height ~ dnorm(mu,sigma),
  mu <- a + b1 * age,
  c(a,b1) ~ dnorm(0,100),
  sigma ~ dunif(0,50)
), data = d1, start = list(a = mean(d1$height), sigma = sd(d1$height),
                           b1 = 0))

model2 <- map(alist(
  height ~ dnorm(mu,sigma),
  mu <- a + b1 * age + b2 * age^2,
  c(a,b1) ~ dnorm(0,100),
  sigma ~ dunif(0,50)
), data = d1, start = list(a = mean(d1$height), sigma = sd(d1$height),
                           b1 = 0, b2 = 0))

model3 <- map(alist(
  height ~ dnorm(mu,sigma),
  mu <- a + b1 * age + b2 * age^2 + b3 * age^3,
  c(a,b1) ~ dnorm(0,100),
  sigma ~ dunif(0,50)
), data = d1, start = list(a = mean(d1$height), sigma = sd(d1$height),
                           b1 = 0, b2 = 0, b3 = 0))

model4 <- map(alist(
  height ~ dnorm(mu,sigma),
  mu <- a + b1 * age + b2 * age^2 + b3 * age^3 + b4*age^4,
  c(a,b1) ~ dnorm(0,100),
  sigma ~ dunif(0,50)
), data = d1, start = list(a = mean(d1$height), sigma = sd(d1$height),
                           b1 = 0, b2 = 0, b3 = 0, b4 = 0))

model5 <- map(alist(
  height ~ dnorm(mu,sigma),
  mu <- a + b1 * age + b2 * age^2 + b3*age^3 + b4*age^4 + b5*age^5,
  c(a,b1) ~ dnorm(0,100),
  sigma ~ dunif(0,50)
), data = d1, start = list(a = mean(d1$height), sigma = sd(d1$height),
                           b1 = 0, b2 = 0, b3 = 0, b4 = 0, b5 = 0))

model6 <- map(alist(
  height ~ dnorm(mu,sigma),
  mu <- a + b1 * age + b2 * age^2 + b3*age^3 + b4*age^4 + b5*age^5 + b6*age^6,
  c(a,b1) ~ dnorm(0,100),
  sigma ~ dunif(0,50)
), data = d1, start = list(a = mean(d1$height), sigma = sd(d1$height),
                           b1 = 0, b2 = 0, b3 = 0, b4 = 0, b5 = 0, b6 = 0))
```

6H1 (2 points)

For question 1, we compare them:

```
compare(model1, model2, model3, model4, model5, model6)
```

```
##          WAIC pWAIC dWAIC weight    SE   dSE
## model4 1926.1   5.7   0.0   0.58 25.51    NA
## model5 1927.7   6.6   1.6   0.26 25.50  1.13
## model6 1928.7   7.8   2.5   0.16 25.00  3.13
## model3 1952.3   5.4  26.1   0.00 24.21 10.86
## model2 2150.0   5.2 223.8   0.00 22.62 26.81
## model1 2395.3   3.3 469.2   0.00 22.94 31.12
```

We see that model 4 performs the best, taking 0.58 of the Akaike weight. This being said, however, the top 3 models here are significantly better than the bottom 3.

6H2 (1 point)

```
age.seq <- seq(from = -2, to = 3, length.out = 30)
mu <- link(model4, data = list(age = age.seq))
```

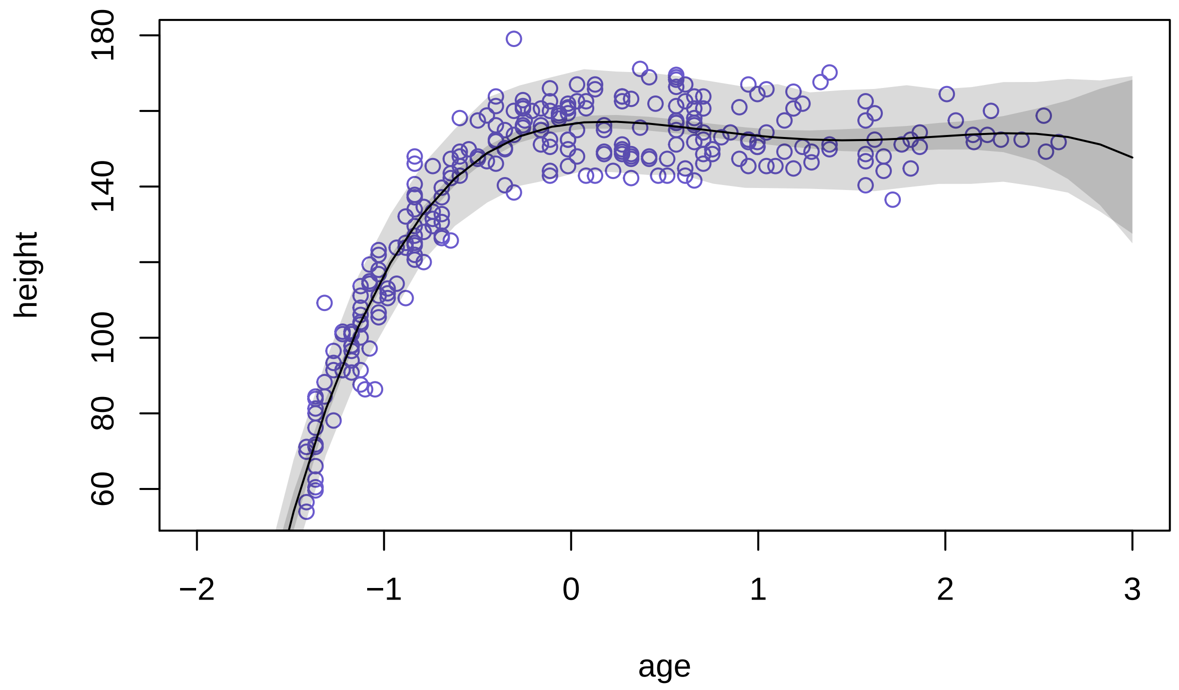
```
## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```
mu_mean <- apply(mu, 2, mean)
mu_pi <- apply(mu, 2, PI, prob = 0.97)
h <- sim(model4, data = list(age = age.seq))
```

```
## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```
height_pi <- apply(h, 2, PI)

plot(height ~ age, d1, col = "slateblue", xlim = c(-2,3))
lines(age.seq, mu_mean)
shade(mu_pi, age.seq)
shade(height_pi, age.seq)
```



If we were to do this for all 6 of the models we made, we would see that the 4th order one hits the “sweet spot” – the other ones are either overfit or underfit. Even this one isn't great, but it's definitely better than the others.

6H3 (1 point)

```
h_ensemble <- ensemble(model4, model5, model6, data = list(age = age.seq))
```

```
## Constructing posterior predictions
```

```
## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

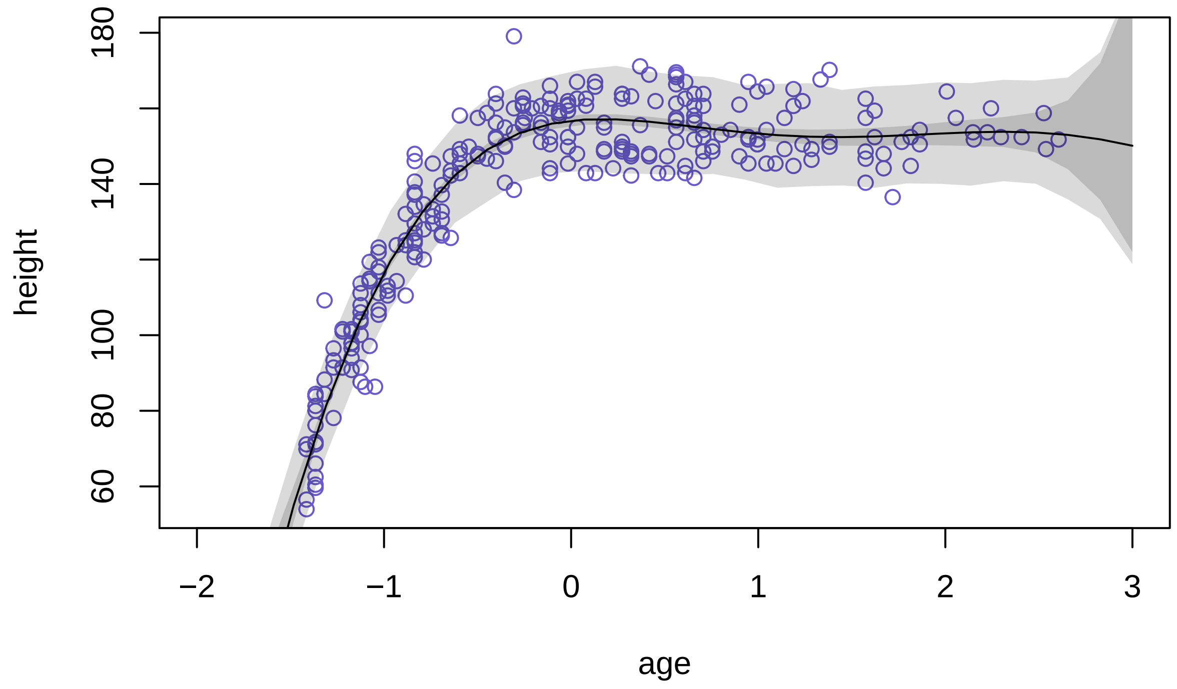
```
## Constructing posterior predictions
```

```
## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```
## Constructing posterior predictions
```

```
## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```r
mu_mean <- apply(h_ensemble$link, 2, mean)
mu_pi <- apply(h_ensemble$link, 2, PI)
height_pi <- apply(h_ensemble$sim, 2, PI)
plot(height ~ age, d1, col = "slateblue", xlim = c(-2, 3))
lines(age.seq, mu_mean)
shade(mu_pi, age.seq)
shade(height_pi, age.seq)
```


height
age