Chapter 6 Homework

Emily Maloney

February 11, 2019

Chapter 6 Homework

```
library(rethinking)
```

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

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

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

```
## rstan (Version 2.18.2, 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)
```

```
## For improved execution time, we recommend calling
## Sys.setenv(LOCAL_CPPFLAGS = '-march=native')
## although this causes Stan to throw an error on a few processors.
```

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

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

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------- tidyverse 1.2.
```

```
## v tibble  2.0.1     v purrr   0.2.5
## v tidyr   0.8.2     v dplyr   0.7.8
## v readr   1.3.1     v stringr 1.3.1
## v tibble  2.0.1     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------------------------------- tidyverse_conflicts()
## x tidyr::extract() masks rstan::extract()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x purrr::map()     masks rethinking::map()
```

```
library(knitr)
```

Easy problems

6E1

The three motivating criteria that define information entropy are:
a) it should be a continuous measure such that it is not sensitive to small changes
b) it should always increase as the number of possible things that could occur (events) increases, because this necessarily makes accuracy more difficult
c) if the number of possible events is greater than two, it should be additive in its approach - i.e. it should sum all of the separate pairs of uncertainties.

6E2

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

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

The entropy of this coin is 0.6108643.

6E3

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

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

The entropy of this die is 1.3762266.

6E4

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

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

The entropy of this die is 1.0986123.

Medium Problems

6M1

All three information criteria, AIC, DIC, and WAIC, take the format of deviance + a penalty term that penalizes the model for having more parameters, but their calculations of each differ depending on how generalized the criterion is. The least general is the Akaike Information Criterion (AIC), whose deviance is -2 * the log likelihood, and the penalty term is simply 2 times the number of parameters. The AIC thus requires the most assumptions: first, that the priors are entirely flat or there is enough data (likelihood) to overwhelm them; second, that the posterior distribution follows a multivariate Gaussian distribution; and third, that the sample size is quite a bit larger than the number of parameters. The Deviance Information Criterion (DIC), relaxes the assumption that the priors have to be flat, now allowing for more informative priors. The DIC calculates deviance as a distribution of deviance, D, and adds the average of that, $\bar{D}$,to the difference between the average and the deviance produced by the posterior average ($\bar{x}$ - $\widehat{D}$), which equates to the penalty term. Finally, the Widely Applicable Information Criterion (WAIC), relaxes the assumption that the posterior distribution follow a multivariate Gaussian distribution in addition to allowing for informative priors, making it the most general. The only assumption that must be met is that the sample size is larger than the number of parameters. In this case, the deviance is -2 times the log-pointwise-predictive-density and the penalty term is the sum of the variances of the likelihoods for each case in the data.

6M2

Model selection means that you use the information criteria that you have calculated to select a single model which you believe performs the best. Model averaging means that you use the weights provided by the information criteria you calculated to produce a new posterior distribution as an amalgation of multiple models weighted according to the probability that each model will make the best predictions on a new data source, in comparison to the other models you estimated. Model averaging allows you to include information about the uncertainty you have about which model actually best represents the world in your final estimations.

6M3

When comparing models, all models must be fit to exactly the same observations, because the information criteria are calculated according to deviance from the observations. Thus, any model that has fewer observations will have a lower information criteria value because it inherently has fewer observations from which to deviate.

6M4

```
data(cars)
m1 <- rethinking::map(alist (
              dist ~ dnorm(mu, sigma),
              mu <- a + b*speed,
              a ~ dnorm(-50, 50),
              b ~ dnorm(0, 20),
              sigma ~ dunif(0,20)),
              data = cars)

precis(m1)
```

```
##         Mean StdDev   5.5% 94.5%
## a     -18.07   6.56 -28.57 -7.58
## b       3.96   0.40   3.32  4.61
## sigma  15.07   1.51  12.66 17.48
```

```
m2 <-  rethinking::map(alist (
              dist ~ dnorm(mu, sigma),
              mu <- a + b*speed,
              a ~ dnorm(-20, 20),
              b ~ dnorm(0, 5),
              sigma ~ dunif(0,20)),
              data = cars)

post1 <- extract.samples(m1, n = 1000)
post2 <- extract.samples(m2, n = 1000)

n_samples <- 1000
ll1 <- sapply(1:n_samples,
             function(s) {
               mu <- post1$a[s] + post1$b[s]*cars$speed
               dnorm(cars$dist, mu, post1$sigma[s], log = T)
             })

ll2 <- sapply(1:n_samples,
             function(s) {
               mu <- post2$a[s] + post2$b[s]*cars$speed
               dnorm(cars$dist, mu, post2$sigma[s], log = T)
             })

pWAIC1 <- sapply(1:nrow(cars), function(i) var(ll1[i,]))
pWAIC2 <- sapply(1:nrow(cars), function(i) var(ll2[i,]))

(pWAIC1 <- sum(pWAIC1))
```

```
## [1] 3.630995
```

```
(pWAIC2 <- sum(pWAIC2))
```

```
## [1] 4.044837
```

As priors get more concentrated, the effective number of parameters decreases, because the variances in the likelihoods are smaller. Since the concentrated priors limit the values that the parameters can take, then the variances among the likelihoods also have to decrease in size.

6M5

Informative priors reduce overfitting because they limit the influence of the data in informing the model. Since overfitting is when the model takes too much information from the data at the risk of not being able to generalize to new observations, informative priors can limit this effect.

6M6

McElreath defines information as the reduction in uncertainty derived from learning an outcome. Taking this into account, overly informative priors can result in underfitting because they limit the amount that a model learns from the data, lessening the information gathered by the model and thus could result in the model not fitting the data well at all and making poor out-of-sample predictions. Overly informative priors mean that you sacrifice the information gained from the data, meaning that your model does not reduce uncertainty or fit the data well.

Hard Problems

```
#loading data in
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,]
```

6H1

```
#estimating all of the models

h1 <- rethinking::map(alist (
              height ~ dnorm(mu, sigma),
              mu <- a + b1*age,
              a ~ dnorm(100, 50),
              b1 ~ dnorm(0, 20),
              sigma ~ dunif(0,50)),
              data = d1)

h2 <- rethinking::map(alist (
              height ~ dnorm(mu, sigma),
              mu <- a + b1*age + b2*I(age)^2,
              a ~ dnorm(100, 50),
              b1 ~ dnorm(0, 20),
```

```
              b2 ~ dnorm(0, 20),
              sigma ~ dunif(0,50)),
              data = d1)

h3 <- rethinking::map(alist (
              height ~ dnorm(mu, sigma),
              mu <- a + b1*age + b2*I(age)^2 + b3*I(age)^3,
              a ~ dnorm(100, 50),
              b1 ~ dnorm(0, 20),
              b2 ~ dnorm(0, 20),
              b3 ~ dnorm(0, 20),
              sigma ~ dunif(0,50)),
              data = d1)

h4 <- rethinking::map(alist (
              height ~ dnorm(mu, sigma),
              mu <- a + b1*age + b2*I(age)^2 + b3*I(age)^3 + b4*I(age)^4,
              a ~ dnorm(100, 50),
              b1 ~ dnorm(0, 20),
              b2 ~ dnorm(0, 20),
              b3 ~ dnorm(0, 20),
              b4 ~ dnorm(0, 20),
              sigma ~ dunif(0,50)),
              data = d1)

h5 <- rethinking::map(alist (
              height ~ dnorm(mu, sigma),
              mu <- a + b1*age + b2*I(age)^2 + b3*I(age)^3 + b4*I(age)^4 + b5*I(age)^5,
              a ~ dnorm(100, 50),
              b1 ~ dnorm(0, 20),
              b2 ~ dnorm(0, 20),
              b3 ~ dnorm(0, 20),
              b4 ~ dnorm(0, 20),
              b5 ~ dnorm(0, 20),
              sigma ~ dunif(0,50)),
              data = d1)

h6 <- rethinking::map(alist (
              height ~ dnorm(mu, sigma),
              mu <- a + b1*age + b2*I(age)^2 + b3*I(age)^3 + b4*I(age)^4 + b5*I(age)^5+ b6*I(age)^6,
              a ~ dnorm(100, 50),
              b1 ~ dnorm(0, 20),
              b2 ~ dnorm(0, 20),
              b3 ~ dnorm(0, 20),
              b4 ~ dnorm(0, 20),
              b5 ~ dnorm(0, 20),
              b6 ~ dnorm(0, 20),
              sigma ~ dunif(0,50)),
              data = d1)

(md <- compare(h1, h2, h3, h4, h5, h6))
```

```
##      WAIC pWAIC dWAIC weight    SE   dSE
## h4 1926.0   5.6   0.0   0.56 25.45    NA
```

```
## h5 1927.5   6.4   1.4   0.28 25.34  0.92
## h6 1928.5   7.7   2.5   0.16 25.08  2.53
## h3 1952.3   5.4  26.2   0.00 24.19 10.84
## h2 2149.7   5.0 223.7   0.00 22.54 26.66
## h1 2395.4   3.4 469.4   0.00 22.98 31.05
```

The model which performs best according to the WAIC is h4, which includes polynomials up to the 4th degree, followed by h6 (with all polynomials up to the sixth degree), h5, then h3, h2, and h1. Interestingly, the weight column shows that h3, h2, and h1 are given 0 weight, indicating that conditioned on the models estimated here, the probability of each of these producing the best estimates is 0. The fourth model, h4, gets the most weight by far, at 0.5601904, and then h6 gets almost a third at 0.2769735, and finally, h5 gets 0.1628349 weight.

6H2

```
N <- 1e4

#PLOTS FOR MODEL 1
preds <-
  as.tibble(MASS::mvrnorm(mu = h1@coef,
                          Sigma = h1@vcov , n = N )) %>%      # rather than extract.samples
  mutate(age = sample(seq(-1.5, 2.6, by = .1), N, replace = T),
         predage = a + b1*age ,                               # line uncertainty
         predheight = rnorm(N, a + b1*age, sigma )) %>%         # data uncertainty
  group_by(age) %>%
  mutate(lb_mu = rethinking::HPDI(predage, prob = .97)[1],
         ub_mu = rethinking::HPDI(predage, prob = .97)[2],
         lb_ht = rethinking::HPDI(predheight, prob = .97)[1],
         ub_ht = rethinking::HPDI(predheight, prob = .97)[2]) %>%
  slice(1) %>%
  mutate(yhat = h1@coef["a"] + h1@coef["b1"] * age) %>%       # yhat for reg line
  select(age, predage, yhat, lb_mu, ub_mu, lb_ht, ub_ht)
```

```
## Warning: `as.tibble()` is deprecated, use `as_tibble()` (but mind the new semantics).
## This warning is displayed once per session.
```

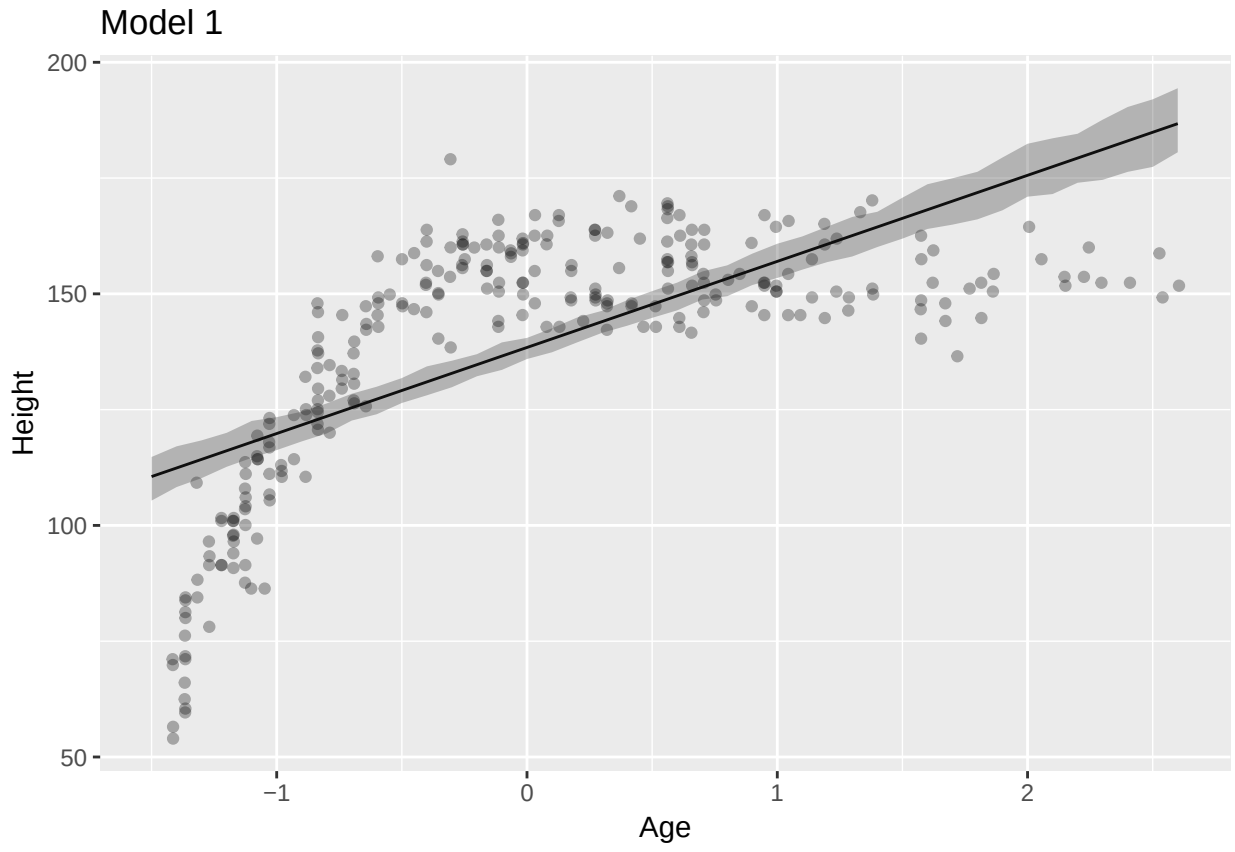
```
#plot
h1_m <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  labs(x = "Age",
       y = "Height",
       title = "Model 1")

h1_mp <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  geom_ribbon(data = preds, aes(ymin = lb_ht, ymax = ub_ht), alpha = .2) +
  labs(x = "Age",
       y = "Height",
       title = "Model 1")

#PLOTS FOR MODEL 2
```

```
preds <-
  as.tibble(MASS::mvrnorm(mu = h2@coef,
                          Sigma = h2@vcov , n = N )) %>%      # rather than extract.samples
  mutate(age = sample(seq(-1.5, 2.6, by = .1), N, replace = T),
         predage = as.integer(a + b1*age + b2*I(age)^2),
         predheight = rnorm(N, a + b1*age + b2*I(age)^2, sigma )) %>%         # data uncertainty
         group_by(age) %>%
         mutate(lb_mu = rethinking::HPDI(predage, prob = .97)[1],
         ub_mu = rethinking::HPDI(predage, prob = .97)[2],
         lb_ht = rethinking::HPDI(predheight, prob = .97)[1],
         ub_ht = rethinking::HPDI(predheight, prob = .97)[2]) %>%
  slice(1) %>%
  mutate(yhat = h2@coef["a"] + h2@coef["b1"] * age + h2@coef["b2"]*I(age)^2) %>%
  select(age, predage, yhat, lb_mu, ub_mu, lb_ht, ub_ht)


#plot
h2_m <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  labs(x = "Age",
       y = "Height",
       title = "Model 2")

h2_mp <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  geom_ribbon(data = preds, aes(ymin = lb_ht, ymax = ub_ht), alpha = .2) +
  labs(x = "Age",
       y = "Height",
       title = "Model 2")

#PLOTS FOR MODEL 3
preds <-
  as.tibble(MASS::mvrnorm(mu = h3@coef,
                          Sigma = h3@vcov , n = N )) %>%      # rather than extract.samples
  mutate(age = sample(seq(-1.5, 2.6, by = .1), N, replace = T),
         predage = as.integer(a + b1*age + b2*I(age)^2+ b3*I(age)^3),
         predheight = rnorm(N, a + b1*age + b2*I(age)^2+ b3*I(age)^3, sigma )) %>%
         group_by(age) %>%
         mutate(lb_mu = rethinking::HPDI(predage, prob = .97)[1],
         ub_mu = rethinking::HPDI(predage, prob = .97)[2],
         lb_ht = rethinking::HPDI(predheight, prob = .97)[1],
         ub_ht = rethinking::HPDI(predheight, prob = .97)[2]) %>%
  slice(1) %>%
  mutate(yhat = h3@coef["a"] + h3@coef["b1"] * age + h3@coef["b2"]*I(age)^2 +
           h3@coef["b3"]*I(age)^3) %>%
  select(age, predage, yhat, lb_mu, ub_mu, lb_ht, ub_ht)

#plot
h3_m <- ggplot(d1, aes(x = age)) +
```

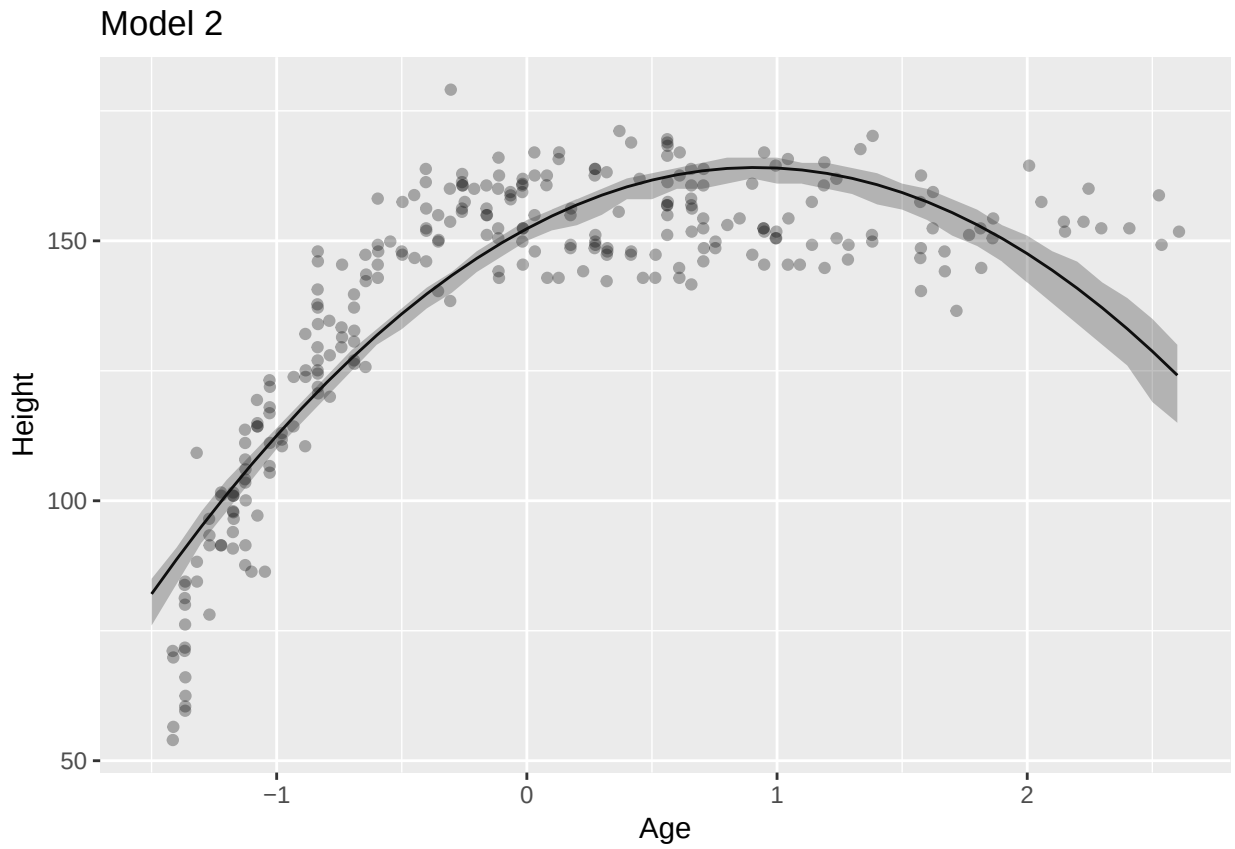
```
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  labs(x = "Age",
       y = "Height",
       title = "Model 3")

h3_mp <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  geom_ribbon(data = preds, aes(ymin = lb_ht, ymax = ub_ht), alpha = .2) +
  labs(x = "Age",
       y = "Height",
       title = "Model 3")

#PLOTS FOR MODEL 4
preds <-
  as.tibble(MASS::mvrnorm(mu = h4@coef,
                          Sigma = h4@vcov , n = N )) %>%      # rather than extract.samples
  mutate(age = sample(seq(-1.5, 2.6, by = .1), N, replace = T),
         predage = as.integer(a + b1*age + b2*I(age)^2+ b3*I(age)^3 + b4*I(age)^4),
         predheight = rnorm(N, a + b1*age + b2*I(age)^2+ b3*I(age)^3+ b4*I(age)^4, sigma )) %>%
         group_by(age) %>%
         mutate(lb_mu = rethinking::HPDI(predage, prob = .97)[1],
         ub_mu = rethinking::HPDI(predage, prob = .97)[2],
         lb_ht = rethinking::HPDI(predheight, prob = .97)[1],
         ub_ht = rethinking::HPDI(predheight, prob = .97)[2]) %>%
  slice(1) %>%
  mutate(yhat = h4@coef["a"] + h4@coef["b1"] * age + h4@coef["b2"]*I(age)^2 +
           h4@coef["b3"]*I(age)^3 + h4@coef["b4"]*I(age)^4) %>%
  select(age, predage, yhat, lb_mu, ub_mu, lb_ht, ub_ht)


#plot
h4_m <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  labs(x = "Age",
       y = "Height",
       title = "Model 4")

h4_mp <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  geom_ribbon(data = preds, aes(ymin = lb_ht, ymax = ub_ht), alpha = .2) +
  labs(x = "Age",
       y = "Height",
       title = "Model 4")

#PLOTS FOR MODEL 5
```

```
preds <-
  as.tibble(MASS::mvrnorm(mu = h5@coef,
                          Sigma = h5@vcov , n = N )) %>%      # rather than extract.samples
  mutate(age = sample(seq(-1.5, 2.6, by = .1), N, replace = T),
         predage = as.integer(a + b1*age + b2*I(age)^2+ b3*I(age)^3 + b4*I(age)^4 +
                                b5*I(age)^5),
         predheight = rnorm(N, a + b1*age + b2*I(age)^2+ b3*I(age)^3+ b4*I(age)^4 +
                              b5*I(age)^5, sigma )) %>%
         group_by(age) %>%
         mutate(lb_mu = rethinking::HPDI(predage, prob = .97)[1],
         ub_mu = rethinking::HPDI(predage, prob = .97)[2],
         lb_ht = rethinking::HPDI(predheight, prob = .97)[1],
         ub_ht = rethinking::HPDI(predheight, prob = .97)[2]) %>%
  slice(1) %>%
  mutate(yhat = h5@coef["a"] + h5@coef["b1"] * age + h5@coef["b2"]*I(age)^2 +
           h5@coef["b3"]*I(age)^3 + h5@coef["b4"]*I(age)^4 + h5@coef["b5"]*I(age)^5) %>%
  select(age, predage, yhat, lb_mu, ub_mu, lb_ht, ub_ht)

#plot
h5_m <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  labs(x = "Age",
       y = "Height",
       title = "Model 5")

h5_mp <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  geom_ribbon(data = preds, aes(ymin = lb_ht, ymax = ub_ht), alpha = .2) +
  labs(x = "Age",
       y = "Height",
       title = "Model 5")

#PLOTS FOR MODEL 6
preds <-
  as.tibble(MASS::mvrnorm(mu = h6@coef,
                          Sigma = h6@vcov , n = N )) %>%      # rather than extract.samples
  mutate(age = sample(seq(-1.5, 2.6, by = .1), N, replace = T),
         predage = as.integer(a + b1*age + b2*I(age)^2+ b3*I(age)^3 + b4*I(age)^4 +
                                b5*I(age)^5 + b6*I(age)^6),
         predheight = rnorm(N, a + b1*age + b2*I(age)^2+ b3*I(age)^3+ b4*I(age)^4 +
                              b5*I(age)^5 + b6*I(age)^6, sigma )) %>%
         group_by(age) %>%
         mutate(lb_mu = rethinking::HPDI(predage, prob = .97)[1],
         ub_mu = rethinking::HPDI(predage, prob = .97)[2],
         lb_ht = rethinking::HPDI(predheight, prob = .97)[1],
         ub_ht = rethinking::HPDI(predheight, prob = .97)[2]) %>%
  slice(1) %>%
  mutate(yhat = h6@coef["a"] + h6@coef["b1"] * age + h6@coef["b2"]*I(age)^2 +
           h6@coef["b3"]*I(age)^3 + h6@coef["b4"]*I(age)^4 + h6@coef["b5"]*I(age)^5 +
```

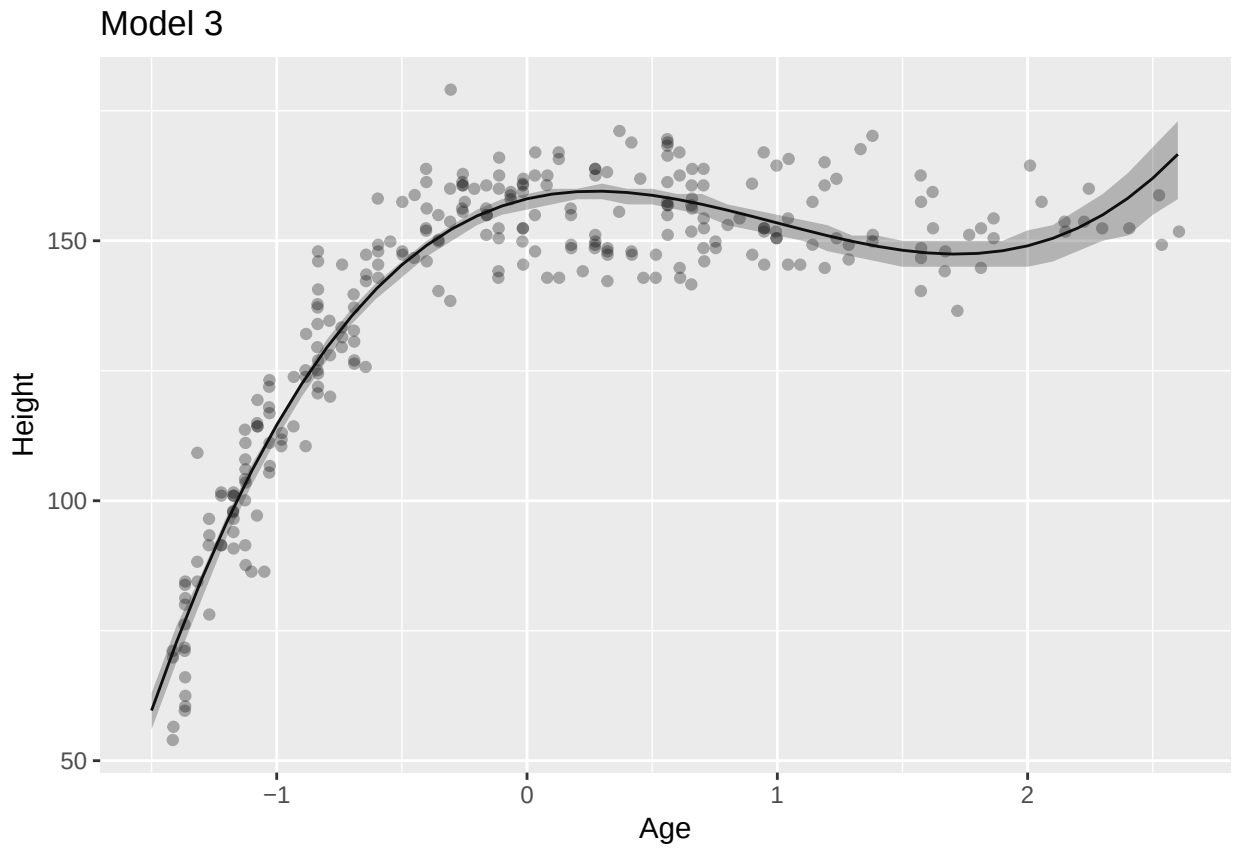
```
          h6@coef["b6"]*I(age)^6) %>%
  select(age, predage, yhat, lb_mu, ub_mu, lb_ht, ub_ht)

#plot
h6_m <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  labs(x = "Age",
       y = "Height",
       title = "Model 6")

h6_mp <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  geom_ribbon(data = preds, aes(ymin = lb_ht, ymax = ub_ht), alpha = .2) +
  labs(x = "Age",
       y = "Height",
       title = "Model 6")

h1_m
```



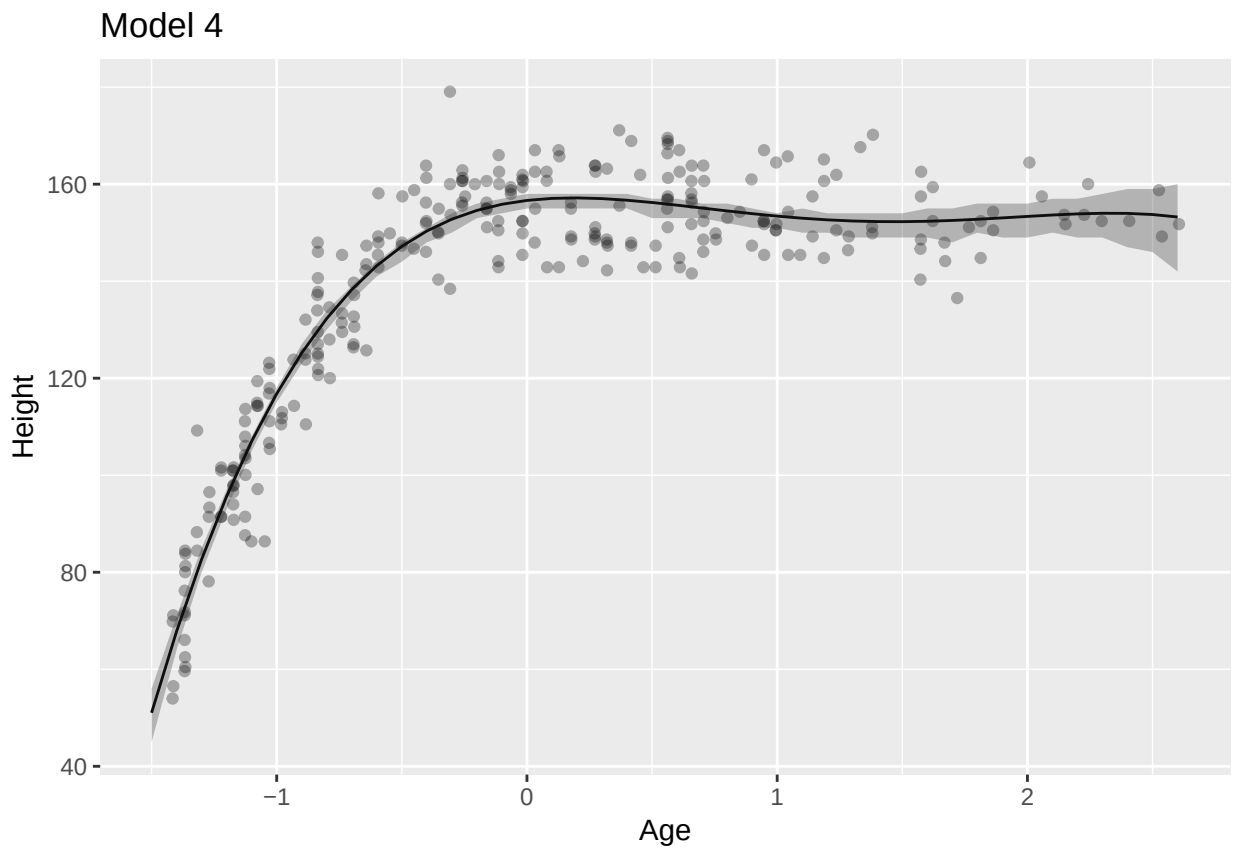

```
h2_m
```



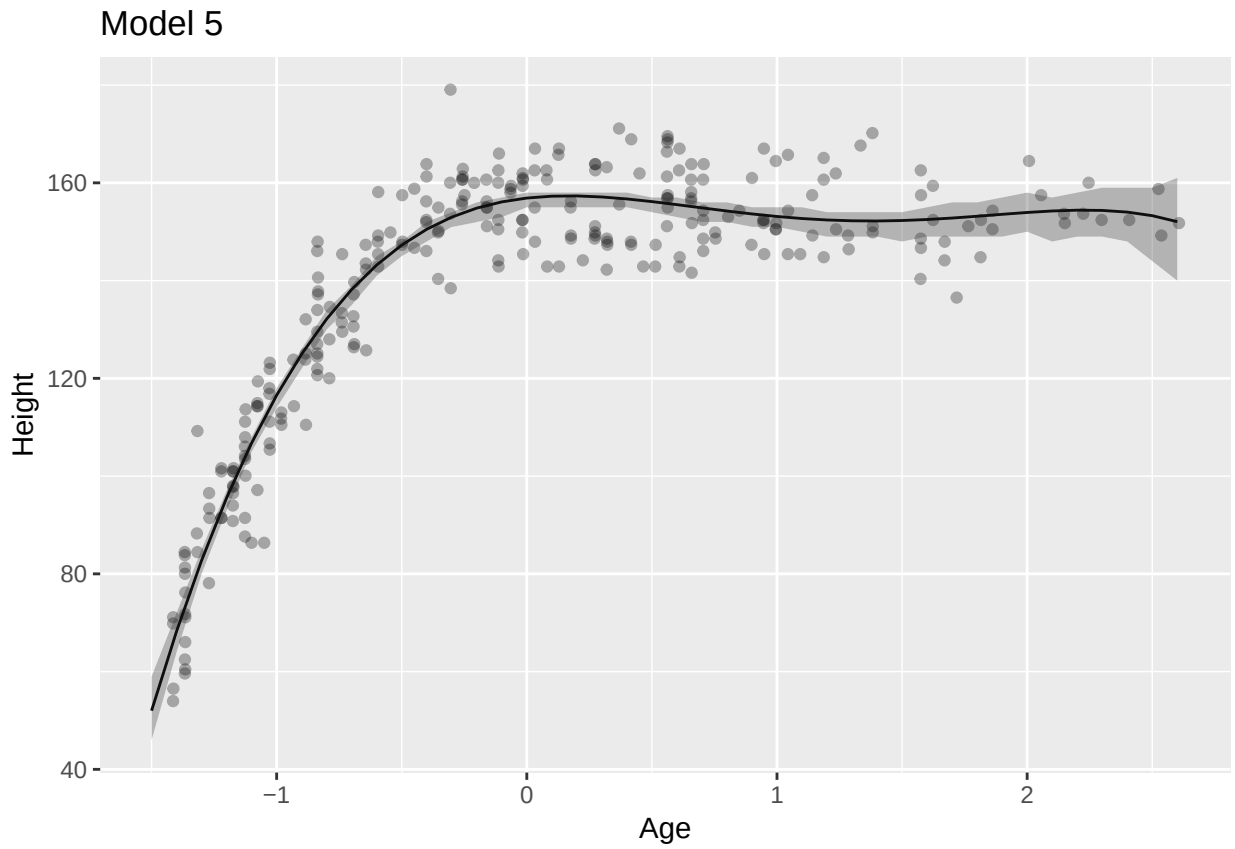

```
h3_m
```



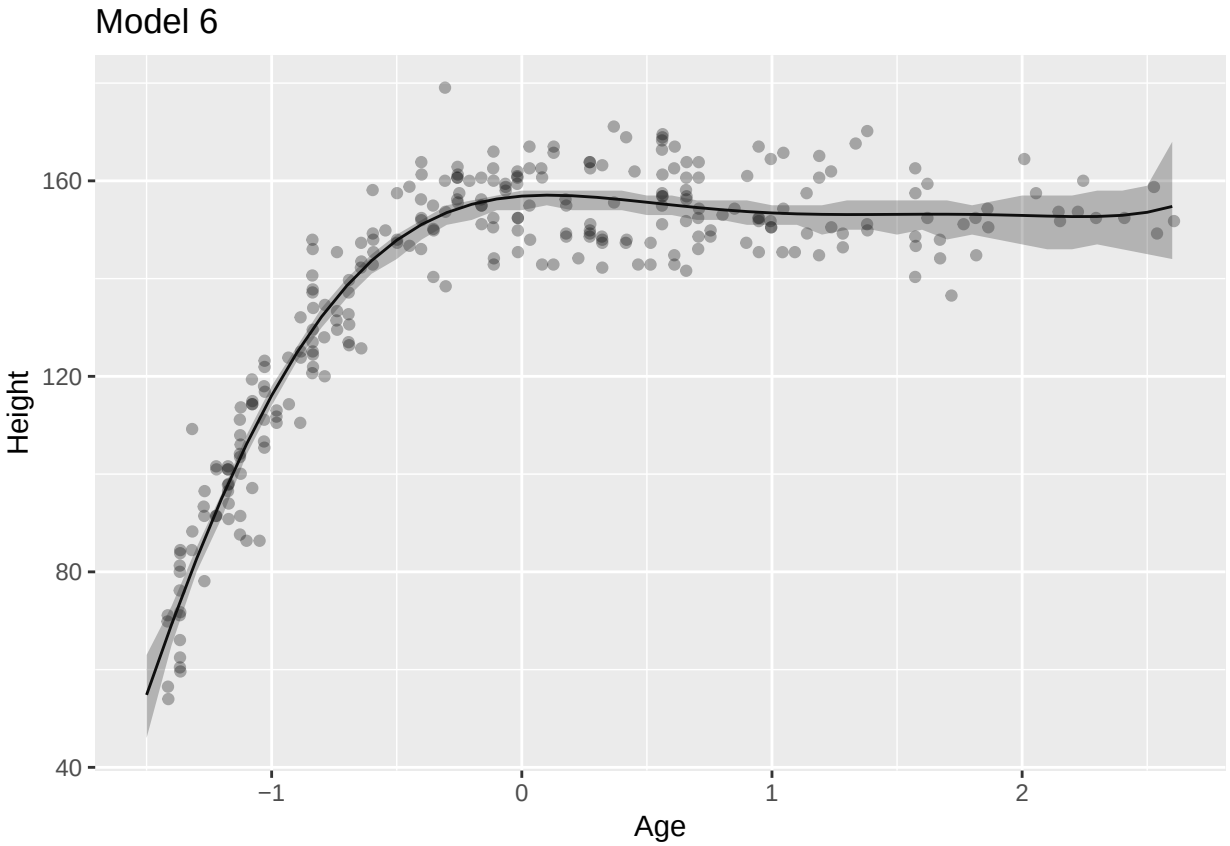

```
h4_m
```




```
h5_m
```



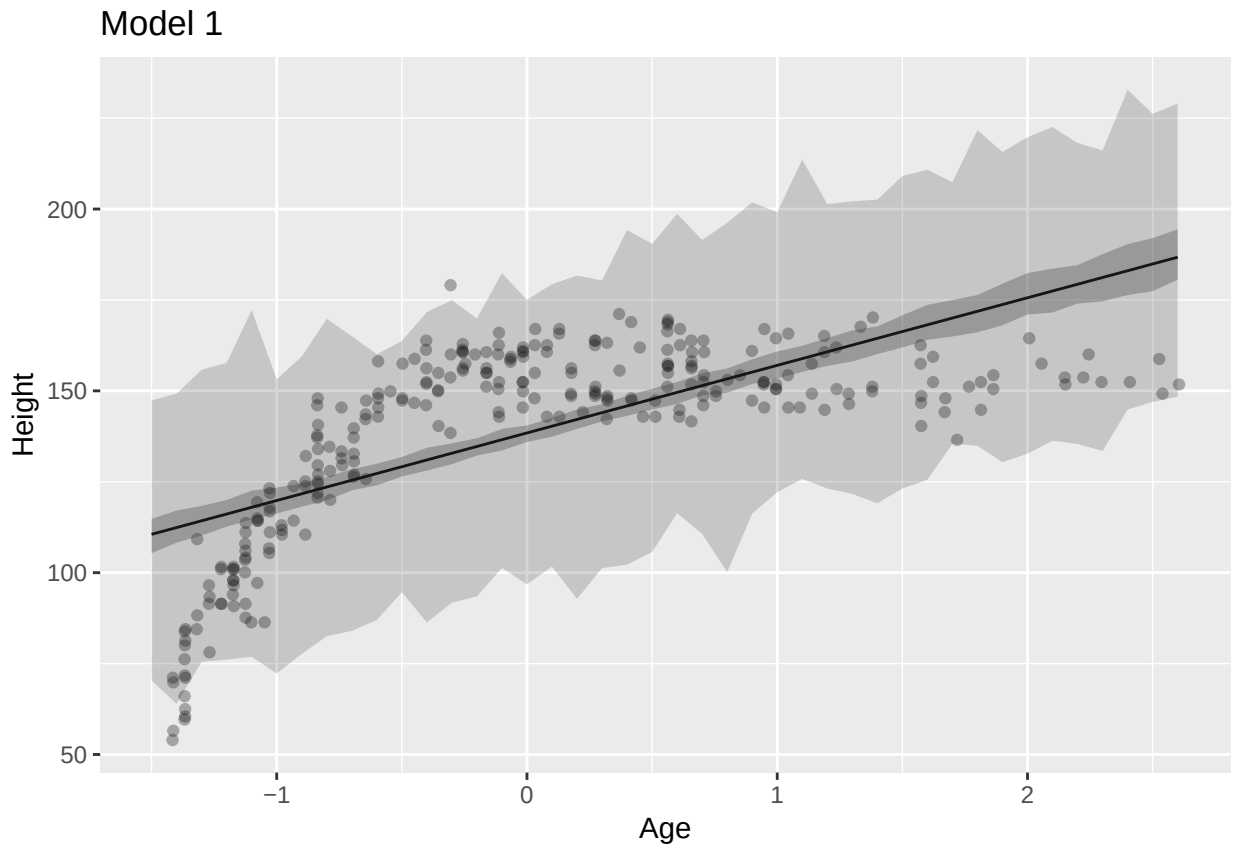

```
h6_m
```



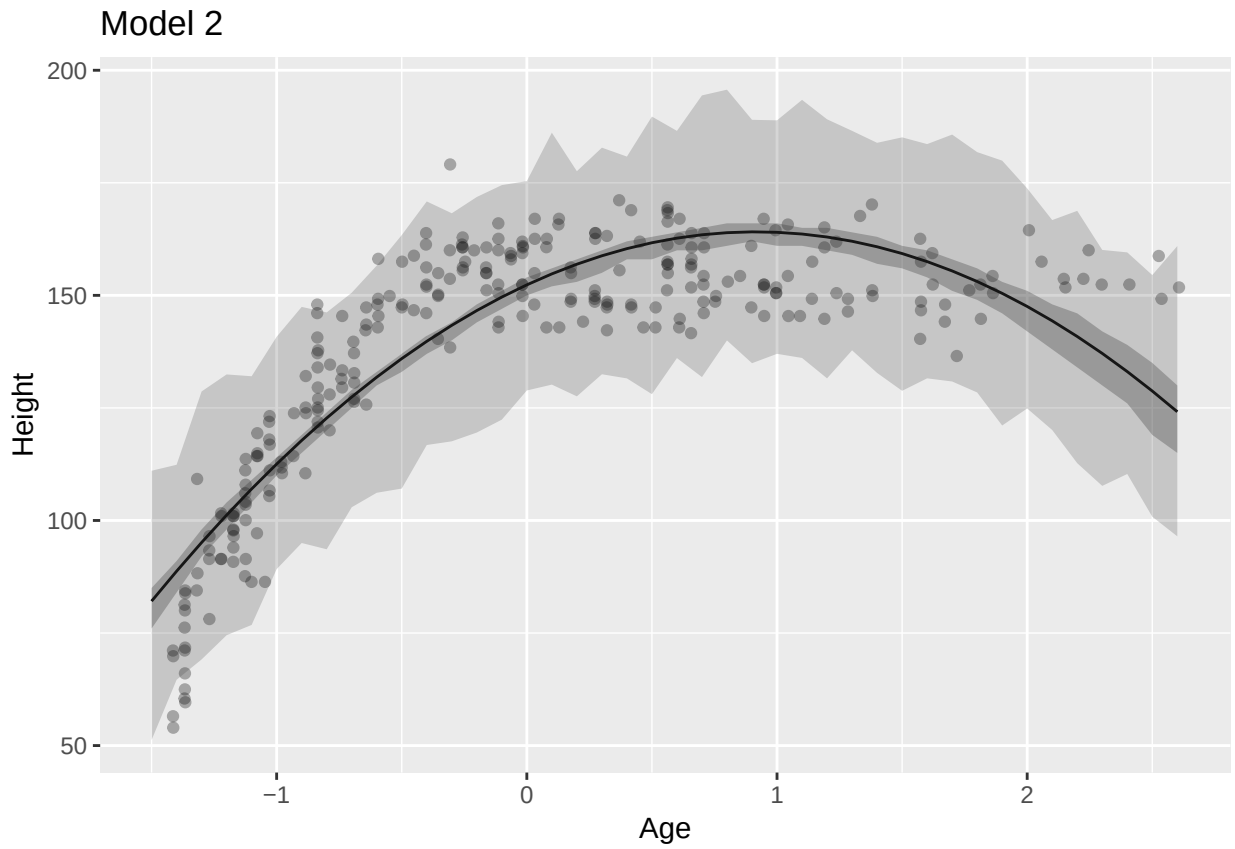
Visualizing the 97% confidence interval of the mean with the superimposed data for each of these models shows how the models with higher degree polynomials have less deviance and thus lower WAIC values, as observed in the previous problem. Particularly, model 1 predicts a continuous linear increase in height according to age, which appears to be a very poor fit with the data, as the data points mostly do not follow the line predicted by the model. Of note is the curved nature of the data's distribution, peaking at slightly below 0 in the age-centered variable. Thus, models 2 and 3 seem to fit a bit better, but still definitely have some of the data points fall above and below the line of best fit as predicted by the model. Model 2 shows a quadratic relationship between age and height, with height increasing as age increases until about age 0.66, and then decreasing as age increases, which does not follow common sense knowledge about how growth works (people do not often shrink). Thus, model 3's inclusion of a cubed term attempts to solve this problem, but then seems to suggest that age and height have a positive relationship between -1.5 and 0.5, slightly negative between 0.5 and 1.5, and then slightly positive beyond age 1.5. This also does not quite follow common sense, because people do not usually have so much fluctuation in height in old age.

Models 4, 5, and 6, look fairly similar in their predictions, just with slight variations in the width of the 97% confidence interval. All three of these predict a sharp increase in height as age goes from -1.5 to about -0.33, and then leveling off until the end of the graph. This tends to follow our instincts about how age should function in relation to height. Additionally, this shows why model 4 had the lowest WAIC - since all three of these models predict approximately the same pattern, model 4 gets the lowest WAIC because it has the least number of effective parameters, i.e. it has a smaller penalty term.

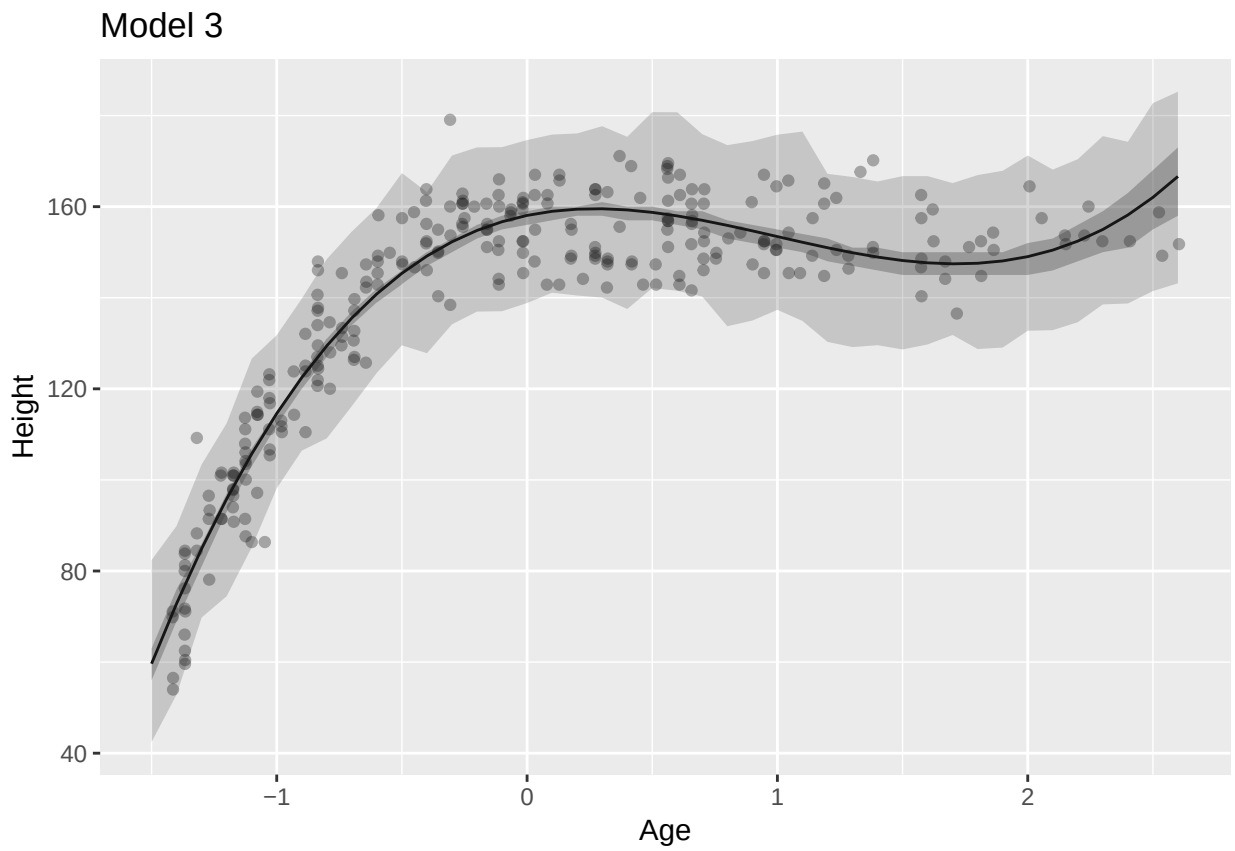
```
#printing plots with predicted data
h1_mp
```



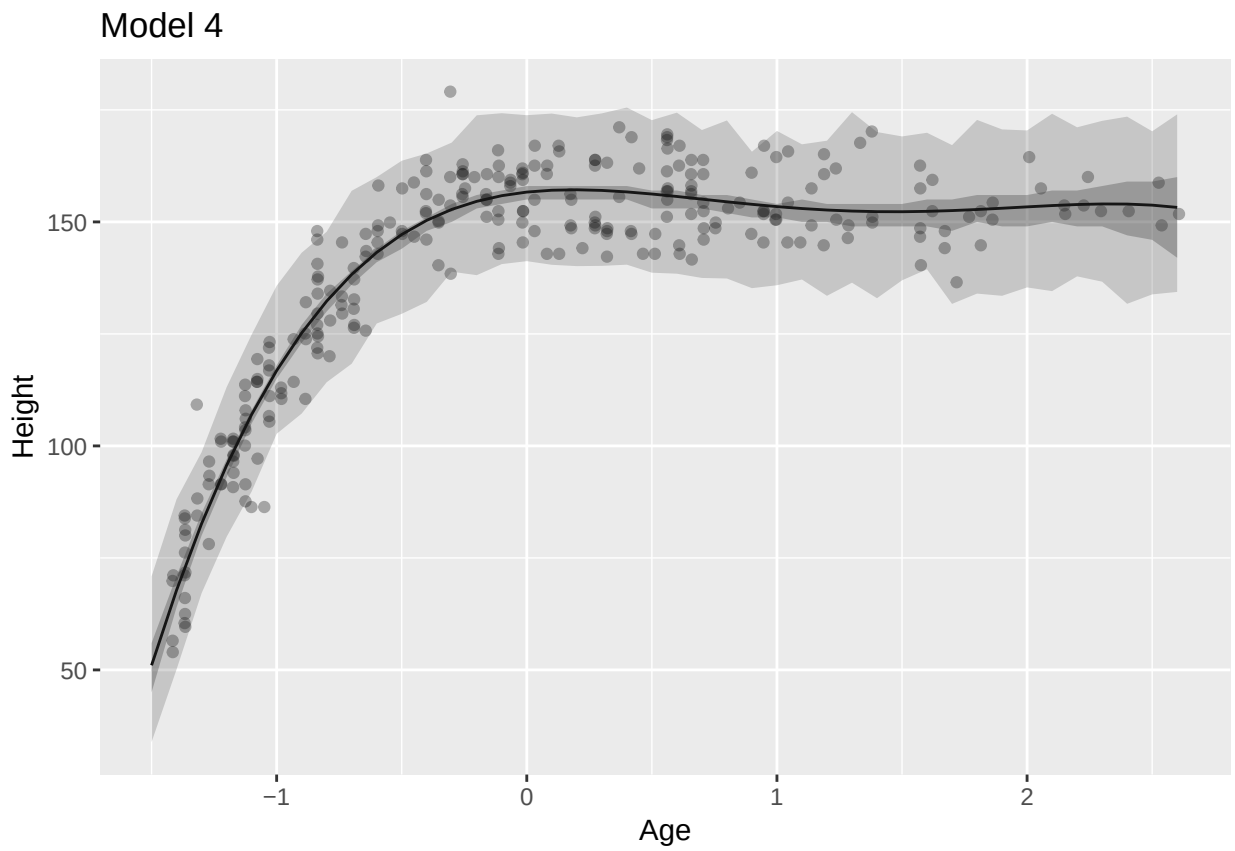

```
h2_mp
```



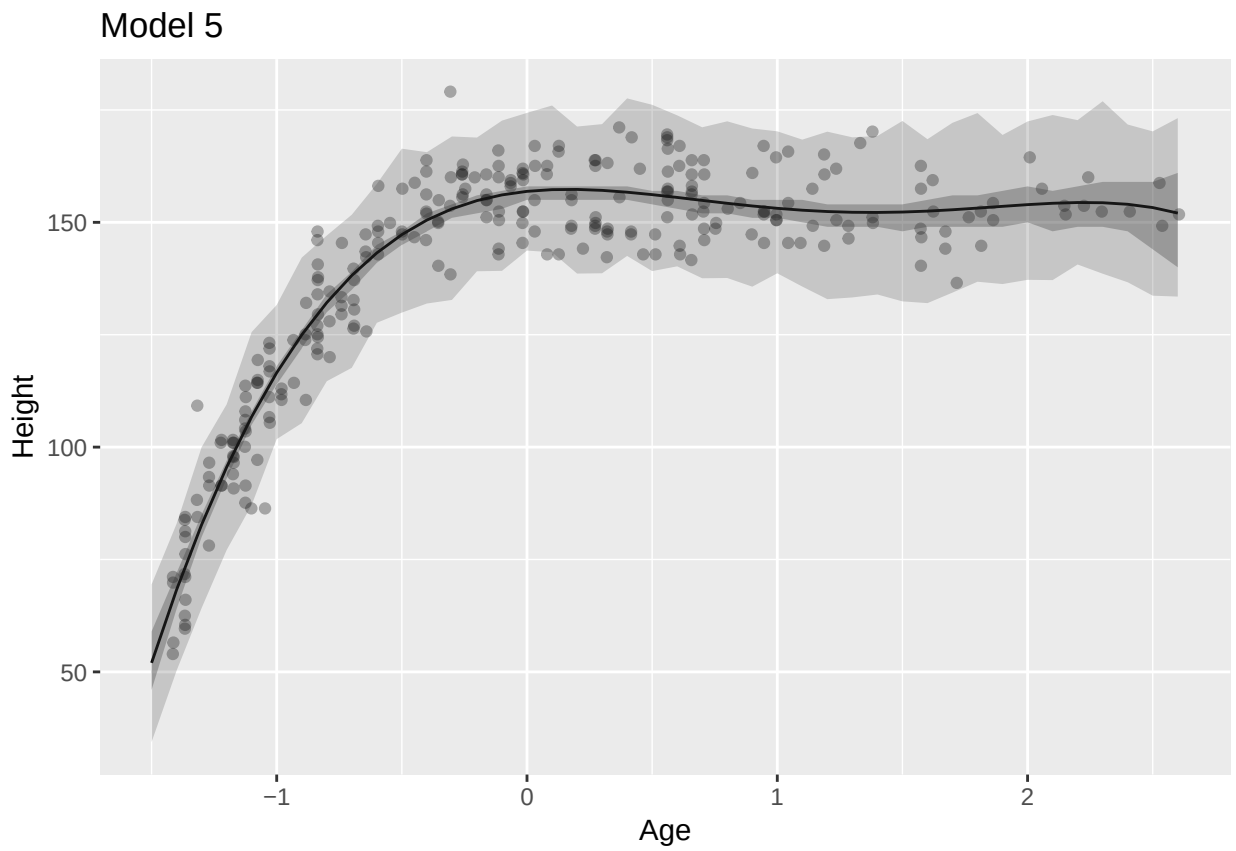

```
h3_mp
```



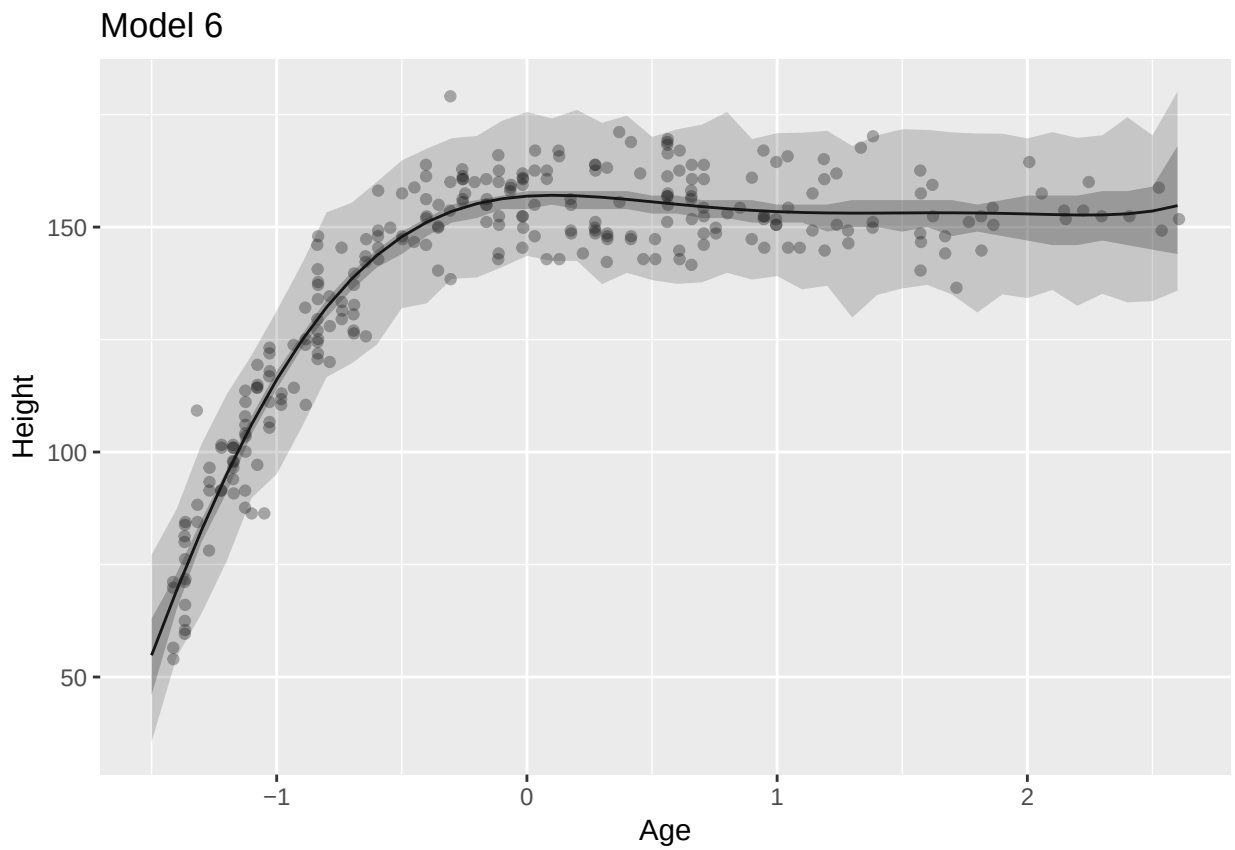

```
h4_mp
```




```
h5_mp
```




```
h6_mp
```



These plots, which show the model averaged predictions in the lighter gray band, differ slightly from the WAIC values. In particular, after model 1, almost all of the models have the large majority of data points inside of the light band, indicating relatively good fit with the data. The difference between what these plots indicate and the WAIC value is that the WAIC takes into account the deviance among the predicted and actual data points, so although models 2 and 3 include the vast majority of the raw data in the confidence interval, these data points fall further away from the mean prediction than in models 4, 5, and 6.

6H4

```
age <- list(age = d2$age)

#deviance for model 1
pred <- link(h1, data = age)
```

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

```
mu <- apply(pred, 2, mean)
dev1 <- -2*sum(dnorm(d2$height, mu, coef(h1)["sigma"], log = T))
```

```r
#deviance for model 2
pred <- link(h2, data = age)
```

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

```r
mu <- apply(pred, 2, mean)
dev2 <- -2*sum(dnorm(d2$height, mu, coef(h2)["sigma"], log = T))

#deviance for model 3
pred <- link(h3, data = age)
```

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

```r
mu <- apply(pred, 2, mean)
dev3 <- -2*sum(dnorm(d2$height, mu, coef(h3)["sigma"], log = T))

#deviance for model 4
pred <- link(h4, data = age)
```

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

```r
mu <- apply(pred, 2, mean)
dev4 <- -2*sum(dnorm(d2$height, mu, coef(h4)["sigma"], log = T))

#deviance for model 5
pred <- link(h5, data = age)
```

```
## [ 100 / 1000 ]
```

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

```
mu <- apply(pred, 2, mean)
dev5 <- -2*sum(dnorm(d2$height, mu, coef(h5)["sigma"], log = TRUE))

#deviance for model 6
pred <- link(h6, data = age)
```

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

```
mu <- apply(pred, 2, mean)
dev6 <- -2*sum(dnorm(d2$height, mu, coef(h6)["sigma"], log = T))

#making table
library(knitr)
dev <- rbind(dev1, dev2, dev3, dev4, dev5, dev6)
rownames(dev) <- c("Model 1", "Model 2", "Model 3", "Model 4", "Model 5", "Model 6")
colnames(dev) <- c("Deviance")
kable(dev)
```

	Deviance
Model 1	2422.160
Model 2	2137.647
Model 3	1932.142
Model 4	1876.547
Model 5	1877.586
Model 6	1876.876

6H5

```
(c <- compare(h1, h2, h3, h4, h5, h6))
```

```
##      WAIC pWAIC dWAIC weight    SE   dSE
## h4 1926.0   5.6   0.0   0.62 25.42    NA
## h5 1928.3   6.8   2.3   0.19 25.56  0.98
## h6 1928.4   7.6   2.4   0.19 25.27  2.56
## h3 1952.8   5.7  26.8   0.00 24.39 11.09
```

```
## h2 2150.1   5.3 224.1   0.00 22.67 26.80
## h1 2395.3   3.3 469.4   0.00 22.95 31.04
```

```
dev1 <- as.data.frame(dev - min(dev))
dev1 <- dev1 %>% arrange(Deviance)
rownames(dev1) <- c("h4", "h6", "h5", "h3", "h2", "h1")
colnames(dev1) <- c("Deviance")

kable(dev1)
```

	Deviance
h4	0.0000000
h6	0.3291878
h5	1.0393350
h3	55.5952295
h2	261.1000518
h1	545.6132894

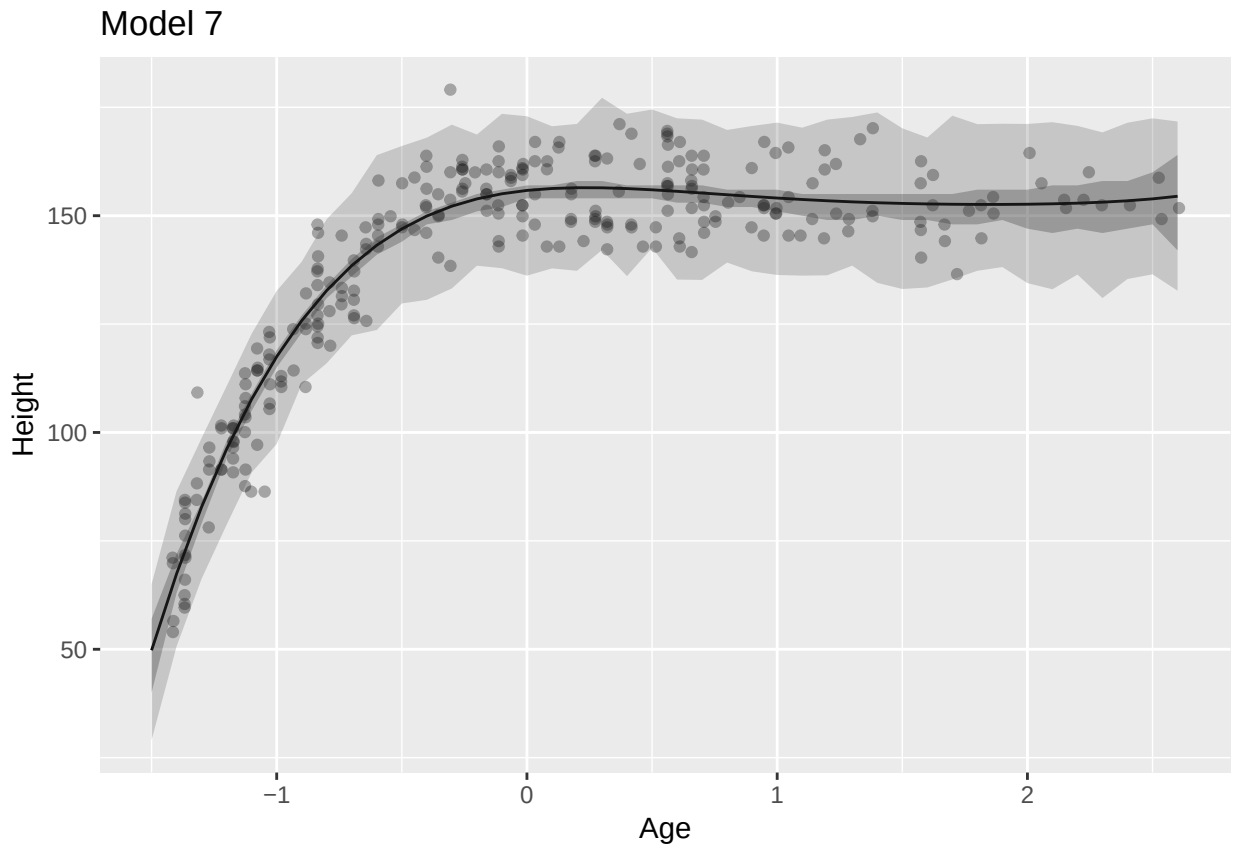
The deviance values produced from the test sample and largely map onto the WAIC results from comparing the models. The main difference is that they do not follow the exact same order. The models arranged in lowest to highest deviance are 4, 5, 6, and then 3, 2, 1, while the models arranged in lowest to highest WAIC are 4, 6, 5, 3, 2, 1. However, the significant differences in deviance and WAIC follow the same pattern, with models 4, 5, and 6 having much lower deviance and WAIC than the other three, and with models 2 and 1 having by far the most deviance and highest WAIC values.

6H6

```
h7 <- rethinking::map(alist (
              height ~ dnorm(mu, sigma),
              mu <- a + b1*age + b2*I(age)^2 + b3*I(age)^3 + b4*I(age)^4 + b5*I(age)^5+ b6*I(age)^6,
              a ~ dnorm(100, 50),
              b1 ~ dnorm(0, 5),
              b2 ~ dnorm(0, 5),
              b3 ~ dnorm(0, 5),
              b4 ~ dnorm(0, 5),
              b5 ~ dnorm(0, 5),
              b6 ~ dnorm(0, 5),
              sigma ~ dunif(0,50)),
              data = d1)

precis(h7)
```

```
##         Mean StdDev   5.5%  94.5%
## a     155.83   0.88 154.43 157.23
## b1      5.97   1.81   3.09   8.86
## b2    -16.58   2.13 -19.99 -13.18
## b3     12.07   2.72   7.72  16.41
## b4     -3.51   1.16  -5.36  -1.65
## b5      0.24   1.04  -1.42   1.90
## b6      0.05   0.33  -0.48   0.58
## sigma   8.20   0.35   7.63   8.76
```

```
#PLOTS FOR MODEL 7
preds <-
  as.tibble(MASS::mvrnorm(mu = h7@coef,
                          Sigma = h7@vcov , n = N )) %>%      # rather than extract.samples
  mutate(age = sample(seq(-1.5, 2.6, by = .1), N, replace = T),
         predage = as.integer(a + b1*age + b2*I(age)^2+ b3*I(age)^3 + b4*I(age)^4 +
                                b5*I(age)^5 + b6*I(age)^6),
         predheight = rnorm(N, a + b1*age + b2*I(age)^2+ b3*I(age)^3+ b4*I(age)^4 +
                              b5*I(age)^5 + b6*I(age)^6, sigma )) %>%
         group_by(age) %>%
         mutate(lb_mu = rethinking::HPDI(predage, prob = .97)[1],
         ub_mu = rethinking::HPDI(predage, prob = .97)[2],
         lb_ht = rethinking::HPDI(predheight, prob = .97)[1],
         ub_ht = rethinking::HPDI(predheight, prob = .97)[2]) %>%
  slice(1) %>%
  mutate(yhat = h7@coef["a"] + h7@coef["b1"] * age + h7@coef["b2"]*I(age)^2 +
           h7@coef["b3"]*I(age)^3 + h7@coef["b4"]*I(age)^4 + h7@coef["b5"]*I(age)^5 +
           h7@coef["b6"]*I(age)^6) %>%
  select(age, predage, yhat, lb_mu, ub_mu, lb_ht, ub_ht)

h7_mp <- ggplot(d1, aes(x = age)) +
  geom_jitter(aes(y = height), alpha = .3) +
  geom_line(data = preds, aes(y = yhat)) +
  geom_ribbon(data = preds, aes(ymin = lb_mu, ymax = ub_mu), alpha = .3) +
  geom_ribbon(data = preds, aes(ymin = lb_ht, ymax = ub_ht), alpha = .2) +
  labs(x = "Age",
       y = "Height",
       title = "Model 7")

h7_mp
```




```
#deviance for model 7
pred <- link(h7, data = age)
```

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

```
mu <- apply(pred, 2, mean)
dev7 <- -2*sum(dnorm(d2$height, mu, coef(h7)["sigma"], log = T))

#compare deviance
cr <- rbind(dev4, dev7)
rownames(cr) <- c("model 4", "model 7")
colnames(cr) <- ("Deviance")
kable(cr)
```

	Deviance
model 4	1876.547
model 7	1875.280

```r
#compare WAIC
compare(h4, h7)
```

```
##      WAIC pWAIC dWAIC weight    SE  dSE
## h4 1926.4   5.8   0.0   0.88 25.47   NA
## h7 1930.5   6.6   4.1   0.12 25.54 2.15
```

In comparison to the model with the lowest WAIC value from before - model 4, which had a WAIC of 1926.1, this model does better with respect to deviance, but worse according to WAIC. This is likely because the WAIC penalizes this model for the extra two parameters it has in comparison to model 4. Essentially this means that the greater precision those extra two parameters afford the model are not great enough to justify adding more parameters to the model.