Chapter 6 Wrap Up?

Stat 341 – Spring 2017

AIC, DIC, WAIC

1. Write definitions for deviance, AIC, DIC, and WAIC.
2. In what ways are the information criteria similar?
3. How do the three information criteria differ from each other?
4. How do the information criteria differ from deviance? Why?
5. Why is it usually better to compare models using DIC or WAIC rather than deviance?

Comparing Models

What does the simulation below do? Be sure to answer the following questions

- What model is being created?
  - What are the parameters of the model?
  - What are the priors on those parameters?
  - Which line gives the likelihood? What does the likelihood say about the model?
- What does the function argument `sigma0` do?
- Explain the use of `logsigma` in this model.

```
sim.map <- function(y, sigma0 = 1) {
  map(
    alist(
      y ~ dnorm(mu, exp(logsigma)),
      mu ~ dnorm(0, sigma0),
      logsigma ~ dnorm(0, 2)
    ),
    data=list(y = y, sigma0 = sigma0) )
}
y <- rnorm(10)
sim.map(y = y)
```

Here are summaries of two data sets

```
favstats(~y1)
```

```
##         min         Q1      median       Q3      max mean sd  n missing
##   -1.304602 -0.5357811 -0.06194462 0.418084 1.819927  0.1  1 10       0
```

```
favstats(~y2)
```

```
##        min       Q1   median       Q3      max mean sd  n missing
##   1.595398 2.364219 2.838055 3.318084 4.719927    3  1 10       0
```

And here are six models fit using those data sets.

```
A1 <- sim.map(y = y1, sigma0 = 0.1)
B1 <- sim.map(y = y1, sigma0 = 1)
C1 <- sim.map(y = y1, sigma0 = 10)
```

```
# different data
A2 <- sim.map(y = y2, sigma0 = 0.1)
B2 <- sim.map(y = y2, sigma0 = 1)
C2 <- sim.map(y = y2, sigma0 = 10)
```

Describe what you would expect to see if you ran this. (It would be good to remind yourself what the rows and columns in the output of `compare()` are.)

```
compare(A1, B1, C1)
```

Describe what you would expect to see if you ran this.

```
lapply(list(A1 = A1, B1 = B1, C1 = C1), coef)
```

Describe what you would expect to see if you ran this.

```
compare(A2, B2, C2)
```

Describe what you would expect to see if you ran this.

```
lapply(list(A2 = A2, B2 = B2, C2 = C2), coef)
```

Averaging Models

Idea: Make predictions based on a mix of the results of several models.

Steps:

1. Fit several models
2. Use WAIC to determine weights of the models
    a. weight will add to 1
    b. "better" models get more weight
3. predict the outcome for each model using `link()` or `sim()`.
4. average these outcomes using the weights.

Implementation: `ensemble()` simplifies much of the work in steps 2–4.

Questions:

1. What are the advantages and disadvantages of model averaging vs model selection?
2. What information does each one lose?

Step 0: Prepare the data

```r
data(milk)
MilkCC <- milk %>% filter(complete.cases(.)) %>%
  mutate(
    neocortex = neocortex.perc / 100
  )
dim(MilkCC)
```

```
## [1] 17  9
```

Step 1: Fit the models

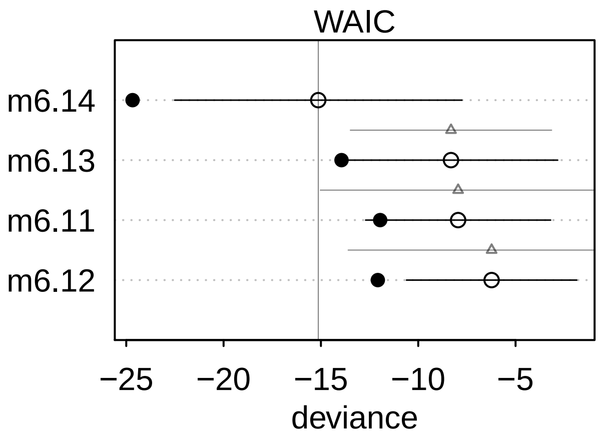
```r
# some starting values to help things converge
a.start <- mean(MilkCC$kcal.per.g)
sigma.start <- log(sd(MilkCC$kcal.per.g))

m6.11 <- map(
  alist(kcal.per.g ~ dnorm(mu, exp(log.sigma)),
        mu <- a),
  data = MilkCC,
  start = list(a = a.start, log.sigma = sigma.start)
)
m6.12 <- map(
  alist(kcal.per.g ~ dnorm(mu, exp(log.sigma)),
        mu <- a + bn * neocortex),
  data = MilkCC,
  start = list(a = a.start, bn = 0, log.sigma = sigma.start)
)
m6.13 <- map(
  alist(kcal.per.g ~ dnorm(mu, exp(log.sigma)),
        mu <- a + bm * log(mass)),
  data = MilkCC,
  start = list(a = a.start, bm = 0, log.sigma = sigma.start)
)
m6.14 <- map(
  alist(kcal.per.g ~ dnorm(mu, exp(log.sigma)),
        mu <- a + bn * neocortex + bm * log(mass)),
  data = MilkCC,
  start = list( a = a.start, bn = 0, bm = 0, log.sigma = sigma.start )
)
```

```r
(milk.comp <- compare(m6.11, m6.12, m6.13, m6.14))
```

```
##        WAIC pWAIC dWAIC weight   SE  dSE
## m6.14 -15.1   4.8   0.0   0.93 7.38   NA
## m6.13  -8.3   2.8   6.8   0.03 5.48 5.17
## m6.11  -8.0   2.0   7.2   0.03 4.75 7.08
## m6.12  -6.2   2.9   8.9   0.01 4.38 7.37
```

```r
plot(milk.comp)
```



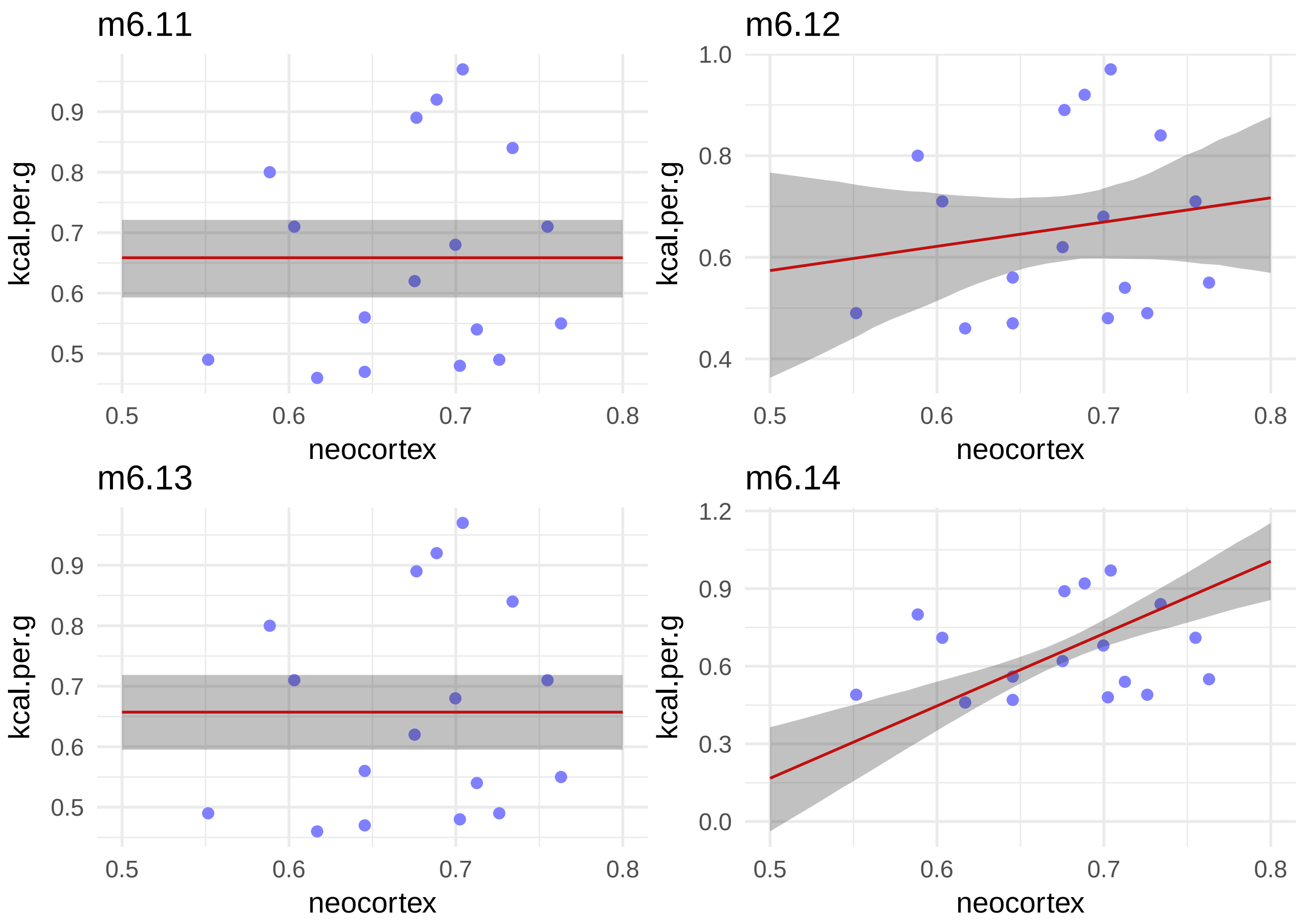
Step 2: Create counterfactual data for our predictions

```
# counterfactual data
Milk.predict <- data_frame(
  neocortex = seq(from = 0.5, to = 0.8, length.out = 30),
  kcal.per.g = 0,
  mass = 4.5  # average mass
)
```

Predictions from individual models

```
# refresh = 0 turns off progress reporting
m6.11.link <- link(m6.11, data = Milk.predict, refresh = 0)
m6.12.link <- link(m6.12, data = Milk.predict, refresh = 0)
m6.13.link <- link(m6.13, data = Milk.predict, refresh = 0)
m6.14.link <- link(m6.14, data = Milk.predict, refresh = 0)
m6.14.sim <- sim(m6.14, data = Milk.predict, refresh = 0)
Milk.predict <-
  Milk.predict %>%
  mutate(
    mu.l611 =  apply(m6.11.link, 2, mean),
    mu.l612 =  apply(m6.12.link, 2, mean),
    mu.l613 =  apply(m6.13.link, 2, mean),
    mu.l614 =  apply(m6.14.link, 2, mean),
    mu.l611.lo = apply(m6.11.link, 2, PI)[1,],
    mu.l611.hi = apply(m6.11.link, 2, PI)[2,],
    mu.l612.lo = apply(m6.12.link, 2, PI)[1,],
    mu.l612.hi = apply(m6.12.link, 2, PI)[2,],
    mu.l613.lo = apply(m6.13.link, 2, PI)[1,],
    mu.l613.hi = apply(m6.13.link, 2, PI)[2,],
    mu.l614.lo = apply(m6.14.link, 2, PI)[1,],
    mu.l614.hi = apply(m6.14.link, 2, PI)[2,],
    mu.s614.lo = apply(m6.14.sim, 2, PI)[1,],
    mu.s614.hi = apply(m6.14.sim, 2, PI)[2,]
    )
```

```
# plot it all
gf_point(kcal.per.g ~ neocortex, data = MilkCC, color = rangi2) %>%
  gf_line(mu.l611 ~ neocortex, data = Milk.predict, color = "red") %>%
  gf_ribbon(mu.l611.lo + mu.l611.hi ~ neocortex, data = Milk.predict) %>%
  gf_labs(title = "m6.11")
gf_point(kcal.per.g ~ neocortex, data = MilkCC, color = rangi2) %>%
  gf_line(mu.l612 ~ neocortex, data = Milk.predict, color = "red") %>%
  gf_ribbon(mu.l612.lo + mu.l612.hi ~ neocortex, data = Milk.predict) %>%
  gf_labs(title = "m6.12")
gf_point(kcal.per.g ~ neocortex, data = MilkCC, color = rangi2) %>%
  gf_line(mu.l613 ~ neocortex, data = Milk.predict, color = "red") %>%
  gf_ribbon(mu.l613.lo + mu.l613.hi ~ neocortex, data = Milk.predict)  %>%
  gf_labs(title = "m6.13")
gf_point(kcal.per.g ~ neocortex, data = MilkCC, color = rangi2) %>%
  gf_line(mu.l614 ~ neocortex, data = Milk.predict, color = "red") %>%
  gf_ribbon(mu.l614.lo + mu.l614.hi ~ neocortex, data = Milk.predict) %>%
  gf_labs(title = "m6.14")
```



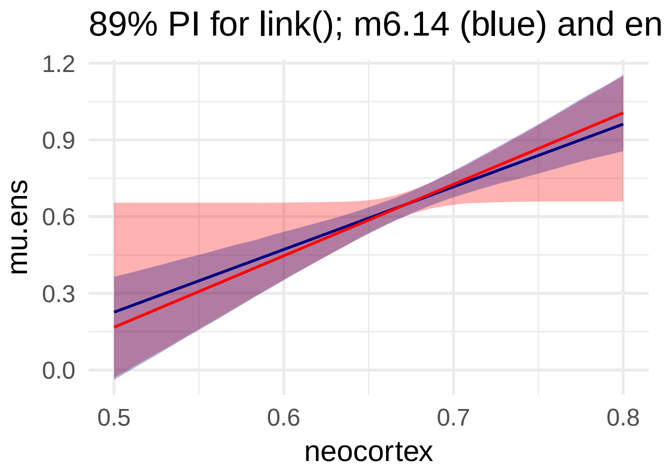
Steps 3 and 4: Average results from component models (using weights)

```
Milk.ensemble <-
  ensemble(m6.11, m6.12, m6.13, m6.14, data = Milk.predict, refresh = 0)
```

```
## Constructing posterior predictions
## Constructing posterior predictions
## Constructing posterior predictions
## Constructing posterior predictions
```

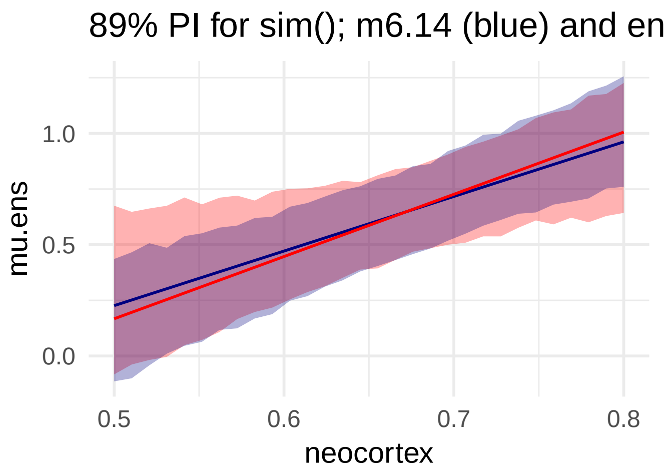
```
Milk.predict <-
  Milk.predict %>%
  mutate(
    mu.ens =  apply(Milk.ensemble$link, 2, mean),
    mu.lens.lo =  apply(Milk.ensemble$link, 2, PI)[1,],
    mu.lens.hi =  apply(Milk.ensemble$link, 2, PI)[2,],
    mu.sens.lo =  apply(Milk.ensemble$sim, 2, PI)[1,],
    mu.sens.hi =  apply(Milk.ensemble$sim, 2, PI)[2,]
    )
```

```
gf_ribbon(mu.lens.lo + mu.lens.hi ~ neocortex, data = Milk.predict, fill = "red")  %>%
  gf_ribbon(mu.l614.lo + mu.l614.hi ~ neocortex, data = Milk.predict, fill = "navy") %>%
  gf_line(mu.ens ~ neocortex, data = Milk.predict, color = "navy") %>%
  gf_line(mu.l614 ~ neocortex, data = Milk.predict, color = "red") %>%
  gf_labs(title = "89% PI for link(); m6.14 (blue) and ensemble (red)")
```



Sim vs Link in Ensemble

```
gf_ribbon(mu.sens.lo + mu.sens.hi ~ neocortex, data = Milk.predict, fill = "red")  %>%
  gf_ribbon(mu.s614.lo + mu.s614.hi ~ neocortex, data = Milk.predict, fill = "navy") %>%
  gf_line(mu.ens ~ neocortex, data = Milk.predict, color = "navy") %>%
  gf_line(mu.l614 ~ neocortex, data = Milk.predict, color = "red") %>%
  gf_labs(title = "89% PI for sim(); m6.14 (blue) and ensemble (red)")
```


89% PI for sim(); m6.14 (blue) and en
mu.ens
neocortex