BDA - Assignment 7 (part 1)

Anonymous

31 10 2019

Contents

Used libraries:

```
library(dplyr)
library(ggplot2)
library(rstan)
library(gdata)
library(bayesplot)
library(aaltobda)
data("drowning")
```

```
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
```

1 Linear model: drowning data with Stan.

1. Spotting two crucial mistakes

NOTE: I will show the full stan code in the latter parts, as I implement the prior. For now, I only point out the problems.

Original code:

```
#data {
#  int<lower=0> N;
#  vector[N] x;
#  vector[N] y;
#  real xpred;
#}
#parameters {
#  real alpha;
#  real beta;
#  real<upper=0> sigma;   HEP! This is problem 1.
#}
#transformed parameters {
#  vector[N] mu;
#  mu = alpha + beta*x;
#}
#model {
#  y ~ normal(mu, sigma);
#}
#generated quantities {
#  real ypred;
#  ypred = normal_rng(mu, sigma);  HEP! This is problem 2.
#}
```

So, **problem 1** is giving an upper bound of zero for deviance (this would result in trying to poll from a distribution with a negative variance). This should be changed to lower bound (parameters block).

Problem 2. is using mu as such. This should be changed to alpha + beta*xpred (generated quantities block). Un-altered, ypred is calculated for whole 'training' data (for each sampled alpha, beta), and eventhough this could be used for other purposes (residuals etc.) the purpose is to get a proper predictive distribution - that is, for a single year (2019).

I will give the fixed code in whole in the 3. task.

2. Figuring out the scale (tau) for prior.

We can use standard normal to poll the necessary scale. I'm using r's qnorm to get the values for the standard case:

```
a <- qnorm(c(0.005,0.995), mean = 0, sd = 1)
a
```

```
## [1] -2.575829  2.575829
```

We can now divide the given quantile information to get the desired deviation:

```
tau <- 69/a[2]
tau
```

```
## [1] 26.78749
```

And just for sanity-check, let's test everything looks as it should:

```
qnorm(c(0.005,0.995), mean = 0, sd = tau)
```

```
## [1] -69  69
```

Everything seems to be in order. And so τ=26.7874893. That being said, this seems somewhat loose constraint on the model (indeed, weakly informative).

3. Fixed Stan implementation.

Finally, here's the proper stan code. I write it on a separate file from rstudio, and then run it separately.

```
write("// I'm writing STAN model from rstudio

      data {
        int<lower=0> N; // number of data points
        vector[N] x; // observation year
        vector[N] y; // observation number of drowned
        real xpred; // prediction year
        real tau;
      }
      parameters {
        real alpha;
        real beta;
        real<lower=0> sigma; //upper bound changed to lower bound
      }
      transformed parameters {
        vector[N] mu;
        mu = alpha + beta*x;
      }
      model {
        beta ~ normal(0, tau); //Here's the added prior
```

```
    y ~ normal(mu, sigma);
  }
  generated quantities {
    real ypred;
    ypred = normal_rng(alpha+beta*xpred, sigma); //changed xpred
  }

  ",

  "stan_model7_1.stan"

  )

#compiles the code
stanc("stan_model7_1.stan")
```

So, I added the prior for beta in model block. Also, real tau in data block (this could have been hard coded as well) and parameter sigma has now a lower bound of zero. Predictions are now done for xpred (2019) as they should (generated quantities block).

I'll run the model to check if everything works.

Here's the drowning data (plus year 2019 for prediction, and the earlier tau for beta's prior):

```
stan_data <- list(N=length(drowning$year), x=drowning$year, y=drowning$drownings,
                  xpred = 2019, tau = tau)

stan_model <- "./stan_model7_1.stan"
```

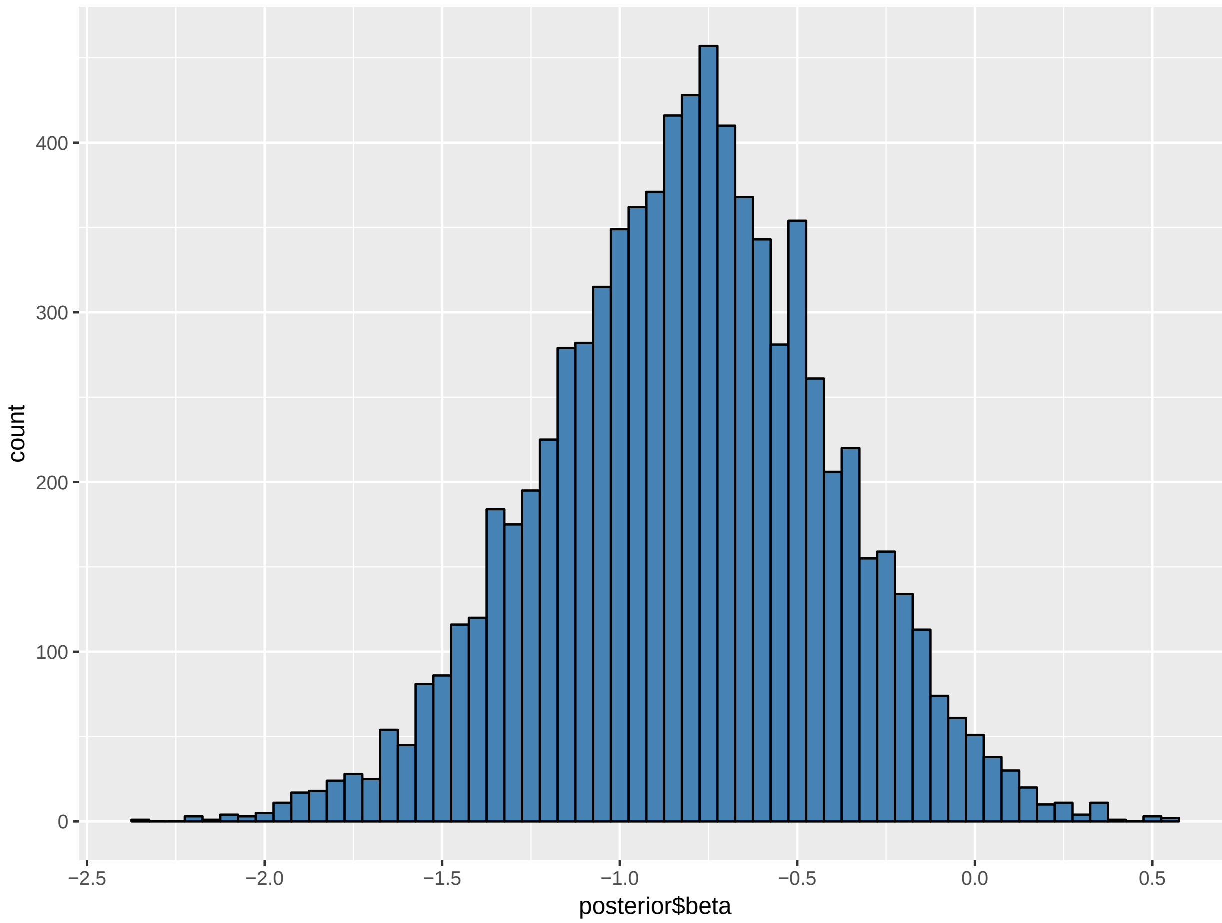
And here's a simple run:

```
fit <- stan(file = stan_model, data = stan_data, warmup = 500, iter = 2500, init = 'random')

# I'll extract the posterior for Rhat and plotting
posterior <- extract(fit)
```

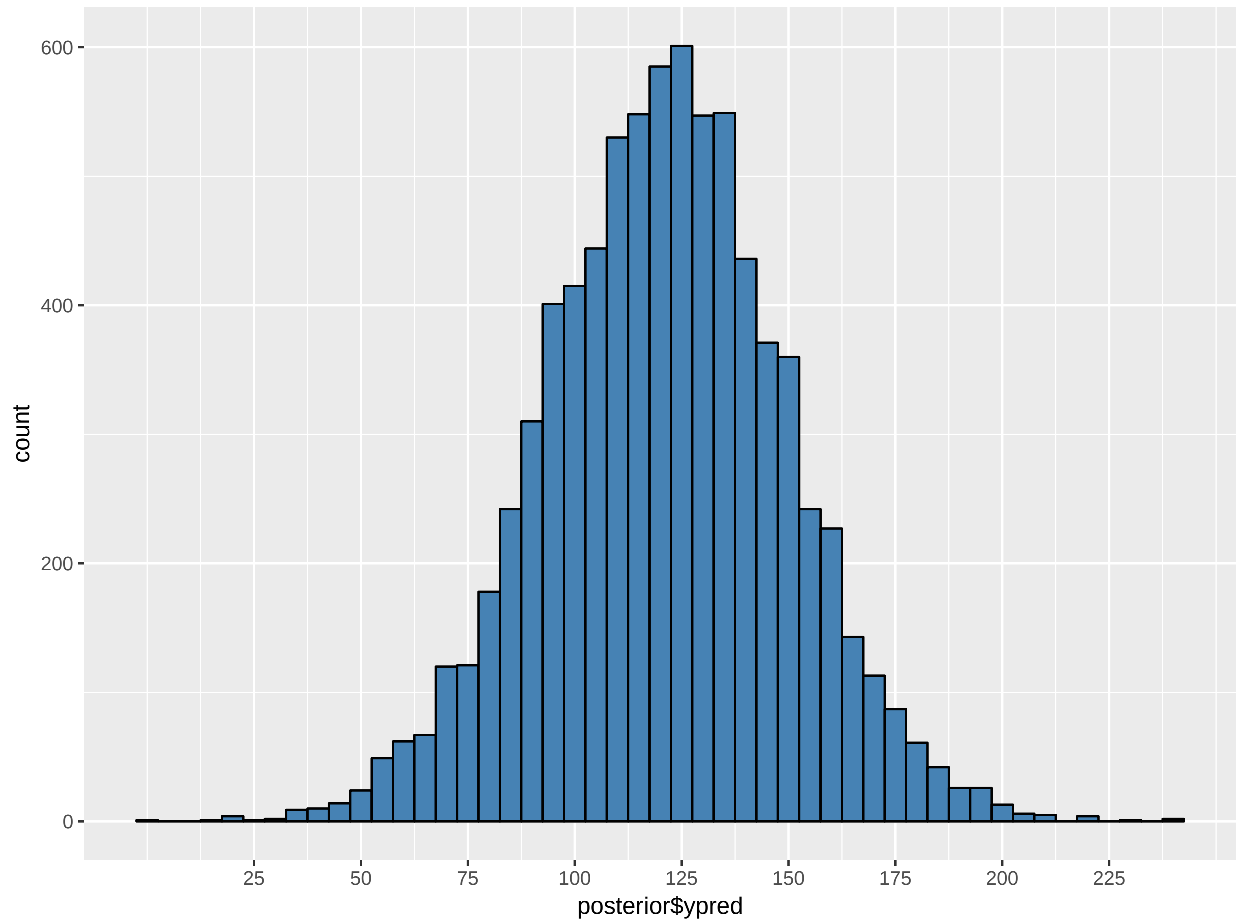
And here are the **Bayesian results**. Histogram for beta:

```
ggplot() +
  geom_histogram(aes(x=posterior$beta),binwidth = 0.05, fill = 'steelblue', color='black') +
  scale_x_continuous(breaks = c(-2.5,-2,-1.5,-1,-.5,0,.5,1))
```



Histogram for predictions:

```
ggplot() +
  geom_histogram(aes(x=posterior$ypred),binwidth = 5, fill = 'steelblue', color='black') +
  scale_x_continuous(breaks = seq(25,225,by = 25))
```



And expected values:

```
mean(posterior$beta)
```

```
## [1] -0.8109086
```

```
mean(posterior$ypred)
```

```
## [1] 120.9438
```

For comparison, a quick linear fit for the data (I only check beta value and prediction):

```
reg.model <- lm(drowning$drownings ~ drowning$year)
reg.model$coefficients[2]
```

```
## drowning$year
##    -0.8176861
```

```
(reg.model$coefficients[1]+ reg.model$coefficients[2]*2019) %>% as.vector()
```

```
## [1] 120.8556
```

As expected with such a loose prior, the results hardly differ. In any case, it seems the number of drowning deaths is decreasing :-)