

BDA - Assignment 8

Anonymous

Contents

Exercise 1	1
Exercise 2	4
Exercise 3	5
Exercise 4	5

List of Tables

1	Expected utility of each machine	4
2	Ranked expected utility of each machine	4

```
library (rstan)
library(loo)
library(tidyverse)
library(knitr)
library(kableExtra)
rstan_options (auto_write = TRUE)
options(mc.cores = parallel::detectCores())
```

Exercise 1

Hierarchical model

As informed by the course staff, applying hyperprios is preferred over uniform priors. Therefore, I will one again use $N \sim (100, 20^2)$ as the weakly informative prior for the means and $\tau \sim Cauchy(0, 10^2)$ as the weakly informative prior for the standard deviations.

Note As I use weakly informative priors, the resulting utilities can deviate from the model solutions a bit.

```
library('aaltobda')
data(factory)

factory_stan = "
data {
  int<lower=0> N;                      // number of data points
  int<lower=0> K;                      // number of groups
  int<lower=1,upper=K> x[N];           // group indicator
  vector[N] y;
```

```

parameters {
  real mu0;                                // prior mean
  real<lower=0> sigma0;                    // prior std
  vector[K] mu;                            // group means
  real<lower=0> sigma;                     // common st.deviation
}
model {
  mu0 ~ normal(100,20);                   // weakly informative prior for hierarchical mean (hyperprior)
  sigma0 ~ cauchy(0, 10);                 // weakly informative prior for hierarchical sigma (hyperprior)
  mu ~ normal(mu0, sigma0);               // weakly informative prior for mean
  sigma ~ cauchy(0,10);                  // weakly informative prior for st.deviation
  y ~ normal(mu[x], sigma);
}
generated quantities {
  real mu7;
  vector[K] ypred;
  real ypred7;
  // posterior od the seventh machine with individual means, common st.deviation and hyperprios
  mu7 = normal_rng(mu0, sigma0);
  for (i in 1:K){
    ypred[i] = normal_rng(mu[i], sigma);
  }
  ypred7 = normal_rng(mu7, sigma);
}
"
factory_hier = list(N = nrow(factory)*ncol(factory),
                     K = ncol(factory),
                     x = rep(1:6, nrow(factory)),      # [1,6] as a bracket
                     y = c(t(factory[,1:6])))

fit_hier = stan(model_code=factory_stan, data=factory_hier, refresh=0)

```

Warning: There were 13 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
 ## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

Warning: Examine the pairs() plot to diagnose sampling problems

```
monitor(fit_hier)
```

```

## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##          Q5     Q50    Q95   Mean    SD   Rhat Bulk_ESS Tail_ESS
## mu0      84.4   93.4  102.6  93.5  5.6    1     2825    2493
## sigma0    4.8   11.1   22.6  12.1  5.7    1     1108    668
## mu[1]     70.7   81.2   92.4  81.2  6.6    1     1997    2136
## mu[2]     92.3  102.4  112.5 102.4  6.2    1     2624    2120
## mu[3]     79.8   89.4   98.8  89.3  5.8    1     3267    2352
## mu[4]     95.1  106.1  117.0 106.0  6.6    1     2098    1748
## mu[5]     80.8   91.0  100.7  90.9  6.0    1     3004    2223
## mu[6]     78.5   88.3   97.5  88.2  5.9    1     2687    2381
## sigma     11.6   14.6   19.2  14.9  2.3    1     2392    1705

```

```

## mu7      71.0   93.4  117.0   93.6 14.5      1    4021   3653
## ypred[1] 54.7   81.5  108.5   81.6 16.6      1    3589   3757
## ypred[2] 74.9   102.0 128.9  102.0 16.4      1    3845   3555
## ypred[3] 63.2   89.5  116.6   89.5 16.3      1    3626   3918
## ypred[4] 79.0   106.5 133.0  106.4 16.5      1    3732   3775
## ypred[5] 63.1   91.2  116.9   90.8 16.5      1    4013   3678
## ypred[6] 62.3   88.3  115.5   88.5 16.2      1    3935   3916
## ypred7   60.2   93.8  127.5   93.8 20.7      1    4123   3774
## lp__    -115.0 -110.0 -107.2 -110.4  2.4      1    1335   1895
##
## For each parameter, Bulk_ESS and Tail_ESS are crude measures of
## effective sample size for bulk and tail quantities respectively (an ESS > 100
## per chain is considered good), and Rhat is the potential scale reduction
## factor on rank normalized split chains (at convergence, Rhat <= 1.05).

utility = function(draws){
  total = 0
  for(i in draws){
    if(i > 85){
      total = total + 94
    }
    else{
      total = total - 106
    }
  }
  total_mean = total/length(draws)
  return(total_mean)
}

draws = as.data.frame(fit_hier, pars = c('ypred[1]', 'ypred[2]', 'ypred[3]',
                                         'ypred[4]', 'ypred[5]', 'ypred[6]'))
machine1 = utility(draws[,1])
machine2 = utility(draws[,2])
machine3 = utility(draws[,3])
machine4 = utility(draws[,4])
machine5 = utility(draws[,5])
machine6 = utility(draws[,6])

df = matrix(c('machine1', machine1,
             'machine2', machine2,
             'machine3', machine3,
             'machine4', machine4,
             'machine5', machine5,
             'machine6', machine6), ncol = 2, byrow = TRUE)
colnames(df) = c('Machine', 'E[U]')
df = as.table(df)
kable(df, 'latex', caption = 'Expected utility of each machine', booktabs = TRUE) %>%
  kable_styling(latex_options = c('striped', 'hold_position'), font_size = 12, full_width = F)

```

Table 1: Expected utility of each machine

	Machine	E[U]
A	machine1	-22.9
B	machine2	65.3
C	machine3	14.55
D	machine4	74.6
E	machine5	23.95
F	machine6	10.3

Exercise 2

```

df = matrix(c('machine1', machine1,
             'machine2', machine2,
             'machine3', machine3,
             'machine4', machine4,
             'machine5', machine5,
             'machine6', machine6), ncol = 2, byrow = TRUE)
colnames(df) = c('Machine', 'E[U]')
df = df[order(df[,2]),]
df = cbind(Rank = c(1,2,3,4,5,6), df)
df = as.table(df)
kable(df, 'latex', caption = 'Ranked expected utility of each machine', booktabs = TRUE) %>%
  kable_styling(latex_options = c('striped', 'hold_position'), font_size = 12, full_width = F)

```

Table 2: Ranked expected utility of each machine

	Rank	Machine	E[U]
A	1	machine1	-22.9
B	2	machine6	10.3
C	3	machine3	14.55
D	4	machine5	23.95
E	5	machine2	65.3
F	6	machine4	74.6

As seen from the ranked utilities (worst to best), we can see that there are three separate clusters in terms of profitability.

1. Machine1 has negative expected utility and is therefore not profitable.
2. Machine6, machine3 and machine5 are all profitable with expected utility values ranging from ≈ 10 to ≈ 24
3. Machine 2 and machine 4 are by far the most profitable ones with expected utilities of ≈ 65 and ≈ 74 , respectively.

Exercise 3

```
utility = function(draws){  
  total = 0  
  for(i in draws){  
    if(i > 85){  
      total = total + 94  
    }  
    else{  
      total = total - 106  
    }  
  }  
  total_mean = total/length(draws)  
  return(total_mean)  
}  
  
draws = as.data.frame(fit_hier, pars = c('ypred7'))  
utility = utility(draws[,1])  
cat('Expected utility of the products of the seventh machine',utility)  
  
## Expected utility of the products of the seventh machine 28.95
```

Exercise 4

Based on the analysis, machine7 is profitable and therefore the company should acquire the machine. Compared to other machines, machine7 is more profitable than machines 3, 5 and 6, but less profitable than machines 2 and 4.