BDA - Assignment 9

Anonymous

```r
options(mc.cores = parallel::detectCores())
library(rstan)
```

```
## Loading required package: StanHeaders

## Loading required package: ggplot2

## rstan (Version 2.19.2, 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.
```

```r
library(ggplot2)
library(aaltobda)
library(loo)
```

```
## This is loo version 2.1.0.
## **NOTE: As of version 2.0.0 loo defaults to 1 core but we recommend using as many as possible. Use t

## **NOTE for Windows 10 users: loo may be very slow if 'mc.cores' is set in your .Rprofile file (see h

##
## Attaching package: 'loo'

## The following object is masked from 'package:rstan':
##
##     loo
```

```r
data("factory")
```

Hierarchical model used

```r
writeLines(readLines("hierarchical.stan"))
```

```
## data {
##   int<lower=0> n;      // No. of data points
##   int<lower=0> g;
##   int<lower=0, upper=g> x[n];
##   vector[n] y;         // data
## }
##
##
## parameters {
##   vector[g] mu;
##   real<lower=0> sigma;
##   real mu_group; // Prior mean
##   real<lower=0> tau; // Prior standard deviation
```

```
## }
##
## model {
##   mu ~ normal(mu_group, tau);
##   y ~ normal(mu[x], sigma);
## }
##
## generated quantities {
##   real mu_seventh;
##   real y_pred[g+1];
##
##   mu_seventh = normal_rng(mu_group, tau);
##   for (i in 1:g) {
##     y_pred[i] = normal_rng(mu[i], sigma);
##   }
##   y_pred[g+1] = normal_rng(mu_seventh, sigma);
## }
```

Data formatting

```r
n = nrow(factory)*ncol(factory)
g = ncol(factory)
x = rep(1:ncol(factory), nrow(factory))
y = c(t(as.matrix(factory)))


input_data = list(n = n,g = g,x = x,y = y)
```

Fitting stan-model

```r
fit_hierarchical = stan(file = "hierarchical.stan", data = input_data)
```

```
## Warning: There were 21 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
```

Production process

The data contains quality measurements of products with the scrap threshold being 85. The price of the product is 200 and production costs per unit is 106. We create a utility function to compute the expected utility of one product of a certain machine.

```r
utility <- function(draws) {
  n <- length(draws)
  utility <- sum((draws >= 85) * 200) / n - 106
  return(utility)
}
```

Hierarchical model

Utility values printed below.

```r
ypred = extract(fit_hierarchical, pars = 'y_pred')
seventh = extract(fit_hierarchical, pars = 'mu_seventh')

df = as.data.frame(fit_hierarchical)

utility(df$`y_pred[1]`)
```

```
## [1] -29.95
```

```r
utility(df$`y_pred[2]`)
```

```
## [1] 65.2
```

```r
utility(df$`y_pred[3]`)
```

```
## [1] 15.6
```

```r
utility(df$`y_pred[4]`)
```

```
## [1] 75.65
```

```r
utility(df$`y_pred[5]`)
```

```
## [1] 18.35
```

```r
utility(df$`y_pred[6]`)
```

```
## [1] 7.5
```

A positive utility means that we expect the machine to be profitable, i.e. it can be operated without loss when taking all costs into consideration. Of the six machins, #1 is the only one which is not profitable. The ranking:

– Worst => 1, 6, 3, 5, 2, 4 <= Best –

Expected utility of the seventh machine

```r
utility(df$`y_pred[7]`)
```

```
## [1] 22.85
```

The 7th machine is expected to be profitable so I would advise the business owner to invest if a) he sees growth in demand and b) he has access to financing.