# BSDA: Assignment 8

November 11, 2023

**Times used for reading and self-study exercise:** 14
**Time used for the assignment:** 10
**Good with the assignment:**
After this assignment I can proudly say that I can evaluate models in Bayesian settings. Another good thing with the assignment is that it was very practical.
**Things to improve in the assignment:**
I would prefer more Stan coding for this Assignment and less interpretation regarding the $\hat{k}-$ values.

## Installing packages and factory data

```
setwd("~/Desktop/stan-demo-files")
library(bsda)
library(loo)
library(rstan)
data("factory")
```

## Task 1

For this task we will fit the Separated, Pooled and Hierarchical model by using the stan-code from Assignment 7. However, since we are going to use loo and psisloo functions later in this assignment we will need to change the Stan-code for it to compute the log-likelihood values for each observation for each posterior draw. This change will be done in the generated quantities block in the Stan code. First we will fit the separated-, then pooled- and lastly the Hierarchical model. For this task, we will use the same hyperpriors, priors and likelihood as in Assignment 7 and recall that we used the following structure for Separated model:

$$
\begin{aligned}
y_{ij} &\sim N\left(\mu_j, \sigma_j\right) \\
\mu_j &\sim N(0, 100) \\
\sigma_j &\sim \text{Inv}-\chi^2(10)
\end{aligned}
$$

and for Pooled model:

$$
\begin{aligned}
y_i &\sim N\left(\mu, \sigma\right) \\
\mu &\sim N(0, 100) \\
\sigma &\sim \text{Inv}-\chi^2(10)
\end{aligned}
$$

and corresponding for Hierarchical model:

$$\begin{aligned}
&\mu_p \sim \mathcal{N}(0, 100) \\
&\sigma_p \sim N^{+}(0, 40) \\
&\theta_j \mid \mu_P, \sigma_P \sim \mathcal{N}\left(\mu_P, \sigma_P\right) \\
&y_{ij} \mid \theta_j \sim \mathcal{N}\left(\theta_j, \sigma_P\right)
\end{aligned}$$

Now when the distributions and the structure for each model has been stated we can fit the model starting with separated model. The stan code for each of the three models can be found in the Appendix. Regarding $\sigma_p$, we have that $\sigma_p \in \mathbf{R}^{+}$ has been stated in the Stan code and $N^{+}(0, 40)$ denotes the half normal distribution on the positive real set.

### Fitting each of the three models

```
# Fitting the Separated model using Stan
data_list <- list(N = nrow(factory), # nobs = 5
                  J = ncol(factory), # ncol = 6
                  y = factory) # the factory data set

fit_separate <- stan(file = "SeperatedModel.stan",
                     data = data_list)

# Fitting the Pooled model using Stan
output <- c(factory$V1, factory$V2,
            factory$V3, factory$V4,
            factory$V5, factory$V6)

data_list2 <- list(y = output,
                   N = nrow(factory) * ncol(factory))

fit_pooled <- stan(file = "PooledModel.stan", data = data_list2)

# Fitting the Hierarchical model using Stan
fit_hier <- stan(file = "HierchModel.stan", data = data_list,
                 chains = 4, iter =   4000)
```

## Task 2

Now when the models has been fitted we want to compute the PSIS-LOO elpd values and the $\hat{k}-$ values for each of the three models. This will be done using the loo()-function in R. Further, after the values has been estimated we will visualize the $\hat{k}-$ values for each model, using a scatter plot, to see how many of these values that is $\hat{k} \leq 0.7$. This is being done to assess the reliability of the PSIS-LOO estimates for each model.

```
par(mfrow = c(3,1))
# LOO for separated model
loo_separate <- loo(fit_separate)
loo_separate
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -138.9  8.9
## p_loo        21.5  5.3
## looic       277.9 17.9
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     18    60.0%   1626
##  (0.5, 0.7]   (ok)        5    16.7%   429
##    (0.7, 1]   (bad)       4    13.3%   37
##    (1, Inf)   (very bad)  3    10.0%   9
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_separate, main = "Separated model")
# Comment: there is five k-values that is above 0.7

# LOO for pooled model
loo_pooled <- loo(fit_pooled)
loo_pooled
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -131.2  5.2
## p_loo         2.5  1.0
## looic       262.5 10.4
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     29    96.7%   1513
##  (0.5, 0.7]   (ok)        1     3.3%   315
##    (0.7, 1]   (bad)       0     0.0%   <NA>
##    (1, Inf)   (very bad)  0     0.0%   <NA>
##
## All Pareto k estimates are ok (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

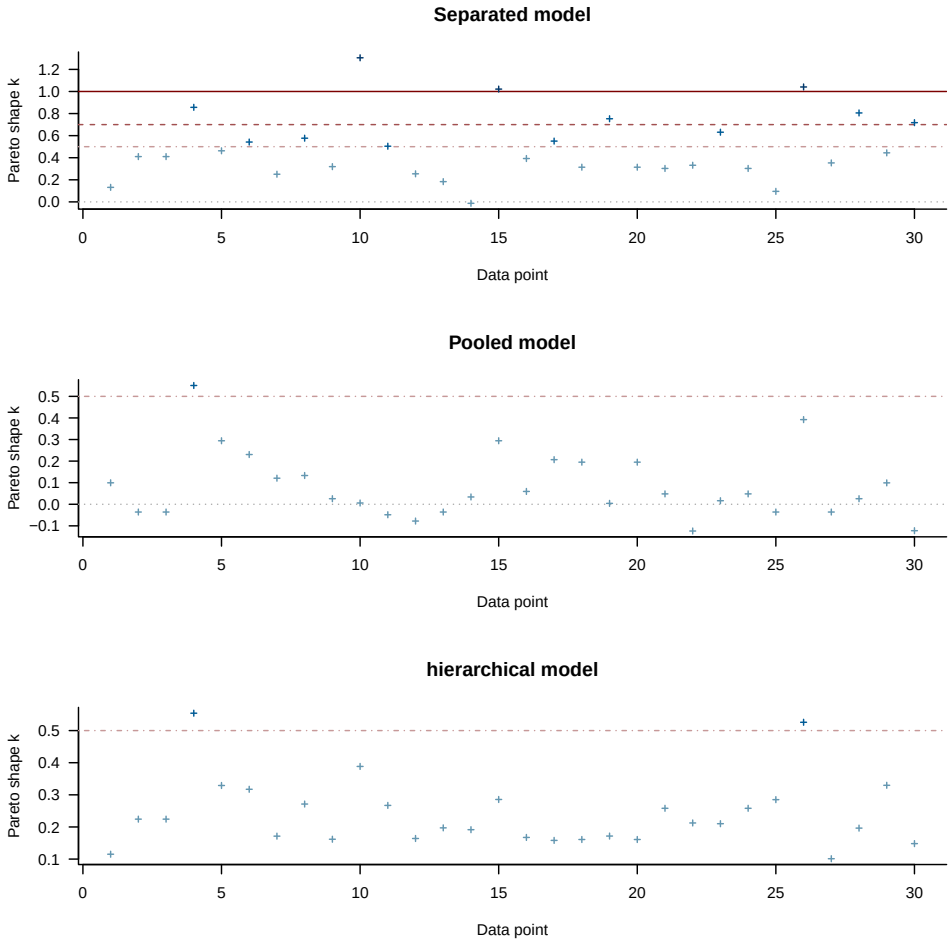
```
plot(loo_pooled, main = "Pooled model")

# LOO for hierarchical model
loo_hierarchical <- loo(fit_hier)
loo_hierarchical
```

```
##
## Computed from 8000 by 30 log-likelihood matrix
```

```
##
##          Estimate   SE
## elpd_loo   -127.0  5.1
## p_loo         6.9  1.8
## looic       254.0 10.2
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     28    93.3%   1563
##  (0.5, 0.7]   (ok)        2     6.7%   558
##    (0.7, 1]   (bad)       0     0.0%   <NA>
##    (1, Inf)   (very bad)  0     0.0%   <NA>
##
## All Pareto k estimates are ok (k < 0.7).
## See help('pareto-k-diagnostic') for details.
```

```
plot(loo_hierarchical, main = "hierarchical model")
```



When looking t the scatter plot for the Separated model we can see that 5 st $\hat{k}-$ values is larger than 0.7. Further, for pooled and hierarchical model both have all $\hat{k} \leq 0.7$.

## Task 3

The effective number of parameters can be calculated as $p_{loo-cv} = lppd - lppd_{loo-cv}$ where lppd stands for the log pointwise predicted density and can be calculated using formula

$$\sum_{i=1}^{n} \log \left( \frac{1}{S} \sum_{s=1}^{S} p\left(y_i \mid \theta^{is}\right) \right)$$

In this case, we will compute the $l\ p\ p\ d$ by using the elpd function in R.

```
# Effective number of parameters for separated model
lppd_sep <- elpd(extract_log_lik(
  fit_separate,
  parameter_name = "log_lik"))$estimates[1]

eff_P_sep <-   lppd_sep  - ( -138.2) # 20.67235
eff_P_sep
```

```
## [1] 20.73787
```

```
# Effective number of parameters for pooled model
lppd_pool <- elpd(extract_log_lik(
  fit_pooled,
  parameter_name = "log_lik"))$estimates[1]

eff_P_pool <-   lppd_pool  - ( -131.2 ) # 2.451017
eff_P_pool
```

```
## [1] 2.425807
```

```
# Effective number of parameters for hierarchical model
lppd_hier <- elpd(extract_log_lik(
  fit_hier,
  parameter_name = "log_lik"))$estimates[1]

eff_P_hier <-   lppd_hier  - ( -127.0 ) # 6.978853
eff_P_hier
```

```
## [1] 6.911867
```

The outputs corresponds to the effective number of parameters for each model. We have that the effective number of parameters for Separated, Pooled and Hierarchical model corresponds to 20.7, 2.5 and 7.0 respectively.

## Task 4

Based on the $\hat{k}-$ values reported in Task 2 we can see that the hierarchical model and pooled model both have that all $\hat{k}-$ values are $\hat{k} \leq 0.7$ whereas separated model has

5 $\hat{k}$ values in the range above 0.7. In general, if all the $\hat{k}$ values is less than or equal to 0.7 then the PSIS estimates can be considered reliable. Hence, the PSIS estimates for hierarchical and pooled model is reliable while the PSIS estimates for separated model is less reliable.

## Task 5

For this task we will look if there is a difference between the models with regards to the $\text{elpd}_{\text{loo-cv}}$. In this case, we will use the model with the highest $\text{elpd}_{\text{loo-cv}}$ and compare that value with the values of the other two models separately. This will be computed using loo-compare function in R.

```
# Looking for the model with the highest elpd value
table <- loo_compare(loo_separate, loo_pooled, loo_hierarchical)
rownames(table) <- c("hierarchical", "pooled", "separated")
table
```

```
##              elpd_diff se_diff
## hierarchical   0.0       0.0
## pooled        -4.3       2.7
## separated    -11.9       5.8
```

The output above shows the difference in elpd for the different models in relation to the model with the highest elpd value which is the hierarchical model. Recall, the loo output from Task 2. This output should be interpreted such as the hierarchical having the highest elpd value while pooled model having 4.1 less compared to hierarchical model. Further, separated model is having 11.2 lesser of a elpd value than hierarchical model.

# Appendix

## Stan code for Separated Model

```
data {
  int < lower = 0 > N; // number of observations
  int < lower = 0 > J; // number of groups in the model
  vector[J] y[N]; // the data set splitted for each group
}

parameters {
  vector[J] mu; // the mean vector, one for each group
  vector<lower =0>[J] sigma; // the sigma vector, only taking on positive values
}

model {

  // prior distributions
  for (j in 1:J) {
    // prior for mu parameter for each machine / group
    mu[j] ~  normal(0, 100);
    // prior for sigma parameter for each machine / group
    sigma[j] ~  inv_chi_square(10);
    // likelihood with respect to mu and sigma parameter
    y[,j] ~ normal(mu[j], sigma[j]);
  }
}

generated quantities {
  real ypred;
  real lppd;
  matrix[N,J] log_lik;
  ypred = normal_rng(mu[6], sigma[6]); // computing the predictive distribution

  for (j in 1:J) {
  for (n in 1:N){
    // computing the log likelihood for each group / machine given the parameters
      log_lik[n,j] = normal_lpdf(y[n,j] | mu[j], sigma[j]);
    }
  }
  lppd = sum(log_lik);
}
```

### Stan code for Pooled Model

```
data {
  // Number of observations where all the groups are combined
  int <lower=0> N;
  // the data set which elements consist of control measurements
  vector[N] y;
}

parameters {
  real mu; // Mean parameter for the pooled machine
  real <lower = 0> sigma; // Variance parameter for the pooled machine
}

model {
  // Prior distributions
  mu ~  normal(0, 100);
  sigma ~  inv_chi_square(10);

  // Likelihood
  y ~  normal(mu, sigma);
}

generated quantities {
  real ypred;
  real mu_upd;
  real log_lik[N];
  real lppd;
  // Computing the predictive distribution for a new machine / group
  ypred = normal_rng(mu, sigma);
  mu_upd = normal_rng(mu, sigma);

  // Computing the log density
  for (n in 1:N){
    log_lik[n] = normal_lpdf(y[n] | mu, sigma);
  }
  lppd = sum(log_lik);
}
```

### Stan code for Hierarchical Model

```
data {
  int < lower = 0 > N; // number of observations
  int < lower = 0 > J; // number of groups in the model
  vector[J] y[N]; // control measurement for all groups
}

parameters {
  vector[J] mu; // the mean vector, one for each group
  real<lower =0> sigma; // the sigma vector, only taking on positive values

  // Hyperparameters, decides the data generating process for the parameters
  real mu_p;
  real <lower =0> sigma_p;
}


model{
  // Hyperprior distributions
  mu_p  ~ normal(0, 100);
  sigma_p  ~ normal(0, 40); // Notice: This is a normal distribution only on the positive re

  // Prior distributions
  sigma  ~ inv_chi_square(10); // df = 10
  for ( j in 1:J){
    mu[j]  ~ normal(mu_p, sigma_p);
  }

  // Likelihood functions
  for ( j in 1:J){
    y[,j]  ~ normal(mu[j], sigma);
  }
}

generated quantities {
  real ypred;
  real mu_upd;
  real y_upd;
  matrix[N,J] log_lik;
  real lppd;

  ypred = normal_rng(mu[6], sigma);
  mu_upd = normal_rng(mu_p, sigma_p);
  y_upd = normal_rng(mu_upd, sigma);

  for ( j in 1:J){
    for (n in 1:N){

      log_lik[n,j] = normal_lpdf(y[n,j] | mu[j], sigma);
    }
  }
```

```
  lppd = sum(log_lik);
}
```