BDA - Assignment 8

Anonymous

5/4/2020

```r
library(tidyverse)
library(aaltobda)
library(rstan)
# stan settings:
source('stan_utility.R') # diagnosis of rhats
options(mc.cores = parallel::detectCores()) #for local computer
rstan_options(auto_write = TRUE) # autosave Stan
# bay settings:
library(loo) #pred. error of MCMC log likelihood
library(gridExtra)
library(bayesplot) #plots of posterior draws (mcmc_hist etc)
library(shinystan) # model paramteres & MCMC simulations
bayesplot_theme_set() #default
SEED <- 48927 # random seed for reproducability
data("factory")
```

In this assignment we will do a model assessment using the leave-one-out cross-validation (LOO-CV) for the factory data from the aaltobda package. In the factory data, the quality of 6 machines are given.

Q1 - Fitting of the models

Given we have 6 machines, our three stan models are:

```r
# display the stanmodels
writeLines(readLines("ex8_separate.stan"))
```

```
## // seperate model for factory data
## 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 {
##   vector[K] mu; // group means
##   vector<lower=0>[K] sigma; // stds of group
## }
## model {
##   y ~ normal(mu[x], sigma[x]);
## }
## generated quantities {
##   real ypred;
##   vector[N] log_lik;
##   ypred = normal_rng(mu[6], sigma[6]);
##   for (i in 1:N)
##     log_lik[i] = normal_lpdf(y[i] | mu[x[i]], sigma[x[i]]);
## }
```

```
writeLines(readLines("ex8_pooled.stan"))
```

```
## // pooled model for factory data
## data {
##   int<lower=0> N; // number of data points
##   vector[N] y; //
## }
## parameters {
##   real mu; // prior means
##   real<lower=0> sigma; // prior std
## }
## model {
##   y ~ normal(mu, sigma);
## }
## generated quantities {
##   real ypred;
##   vector[N] log_lik;
##   ypred = normal_rng(mu, sigma);
##   for (i in 1:N)
##     log_lik[i] = normal_lpdf(y[i] | mu, sigma);
## }
```

```
writeLines(readLines("ex8_hierarchical.stan"))
```

```
## // hierarchical model for factory standard
## 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 stds
## }
## model {
##   mu0 ~ normal(90, 15); // weakly informative prior
##   sigma0 ~ cauchy(0,4); // weakly informative prior
##   mu ~ normal(mu0, sigma0); // population prior with unknown parameters
##   sigma ~ cauchy(0,4); // weakly informative prior
##   y ~ normal(mu[x] , sigma);
## }
## generated quantities {
```

```
##   real ypred;
##   real mu7;
##   vector[N] log_lik;
##   ypred = normal_rng(mu[6], sigma);
##   mu7 = normal_rng(mu0, sigma0);
##   for (i in 1:N)
##     log_lik[i] = normal_lpdf(y[i] | mu[x[i]], sigma);
## }
```

Fitting of the seperate model:

Assumptions:

- Priors are uniform
- There are 6 machines
- Each machine (j) has unrelated means μ_j and standard deviations σ_j

```r
# creating data
d_separate <-list(N = 30,
                  K = 6,
                  x = rep(1:ncol(factory), nrow(factory)),
                  y = c(t(factory)))
# fitting
fit_separate <- stan(file="ex8_separate.stan",
                     data = d_separate, seed = SEED)
```

Fitting of the pooled model:

Assumptions:

- Priors are uniform
- The machines equals only one machine, where all the data comes from
- The one machince has one mean μ and one standard deviation σ

```r
# creating data
d_pooled <- list(N = 30,
                 y = c(t(factory)))
# fitting
fit_pooled <- stan(file = "ex8_pooled.stan",
                   data = d_pooled, seed = SEED)
```

Fitting of the hierarchical model:

Assumptions:

- Prior distribution follows: $\mu \sim \mathcal{N}(\mu_{\mathbf{0}}, \sigma_{\mathbf{0}})$
- Equal standard deviation for all machines σ

```r
# creating data
d_hierarchical <-list(N = 30,
                      K = 6,
                      x = rep(1:ncol(factory), nrow(factory)),
                      y = c(t(factory)))
# fitting
fit_hierarchical <- stan(file="ex8_hierarchical.stan",
                         data = d_hierarchical, seed = SEED)
```

```
## Warning: There were 39 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

## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

Q2

For model checking and comparison of models, we will do a Pareto smoothed importance-sampling leave-one-out cross-validation (PSIS-LOO) to compute the expected log predictive density (elpd) values and $\hat{k}$-values. We use the functions from the loo package for this and for a diagnostic plot, visualizing the $\hat{k}$-values. The horizontal lines in the plot helps determing if the $\hat{k}$-values are good or not. A value <5 is considered good, <0.7 is ok and >0.7 is bad.

Separate

```r
# seperate
log_lik_separate <- extract_log_lik(fit_separate,
                                    merge_chains = FALSE)
r_eff_separate <- relative_eff(exp(log_lik_separate))
loo_separate <- loo(log_lik_separate,
                    r_eff = r_eff_separate)
```

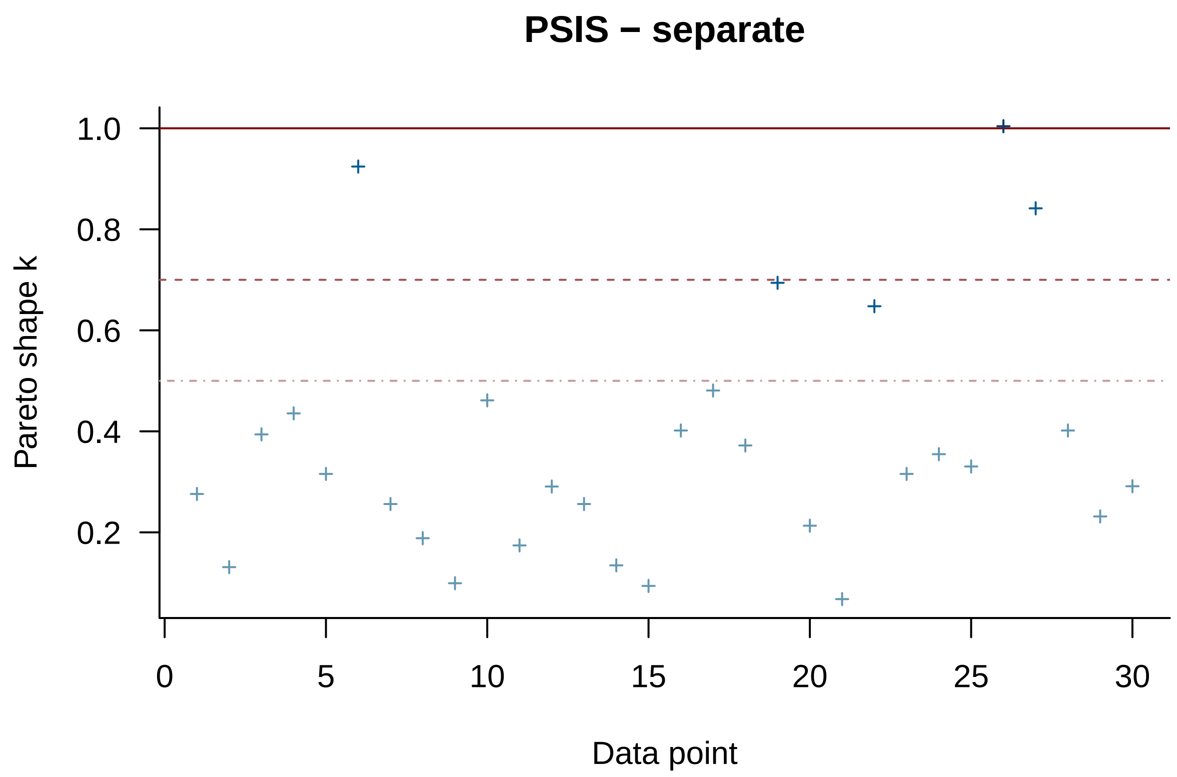
```
## Warning: Some Pareto k diagnostic values are too high. See help('pareto-k-diagnostic') for details.
```

```r
print(loo_separate)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -132.6 3.2
## p_loo         9.9 1.1
## looic       265.2 6.3
## ------
## Monte Carlo SE of elpd_loo is NA.
##
```

```
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     25    83.3%   442
##  (0.5, 0.7]   (ok)        2     6.7%   274
##    (0.7, 1]   (bad)       2     6.7%   77
##    (1, Inf)   (very bad)  1     3.3%   64
## See help('pareto-k-diagnostic') for details.
```

```
# visulazing
plot(loo_separate, diagnostic = c("k", "n_eff"),
     label_points = FALSE, main = "PSIS - separate")
```



For the separate model we get:

- PSIS-LOO : -132.6
- $\hat{k}$: Pareto k diagnostic values are slightly high (10% is above 0.7.

Pooled

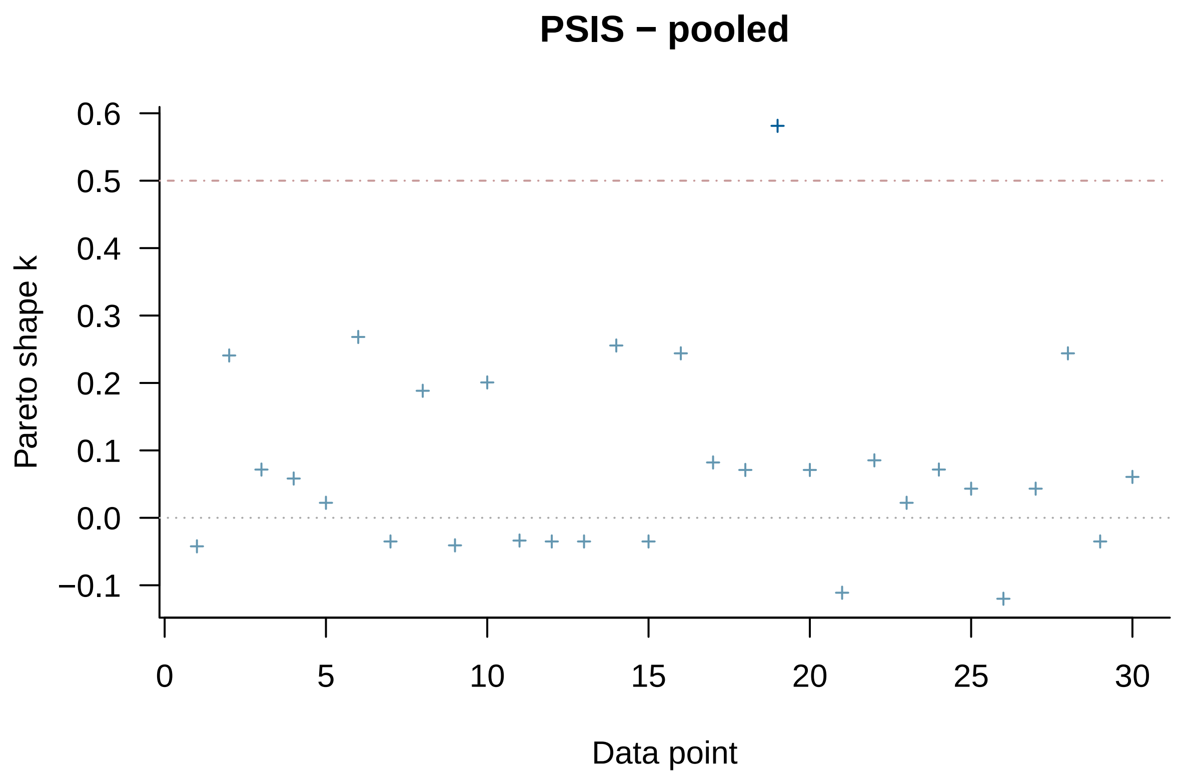
```
# pooled
log_lik_pooled <- extract_log_lik(fit_pooled,
                                  merge_chains = FALSE)
r_eff_pooled <- relative_eff(exp(log_lik_pooled))
loo_pooled <- loo(log_lik_pooled,
                  r_eff = r_eff_pooled)
```

```
## Warning: Some Pareto k diagnostic values are slightly high. See help('pareto-k-diagnostic') for deta
```

```
print(loo_pooled)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -131.0 4.3
## p_loo         2.1 0.8
## looic       262.1 8.6
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     29    96.7%   1896
##  (0.5, 0.7]   (ok)        1     3.3%   510
##    (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.
```

```
# visualizing:
plot(loo_pooled, diagnostic = c("k", "n_eff"),
     label_points = FALSE, main = "PSIS - pooled")
```



For the pooled model we get:

- PSIS-LOO: -131.0
- $\hat{k}$: All Pareto k estimates are ok (k < 0.7)

hierachical

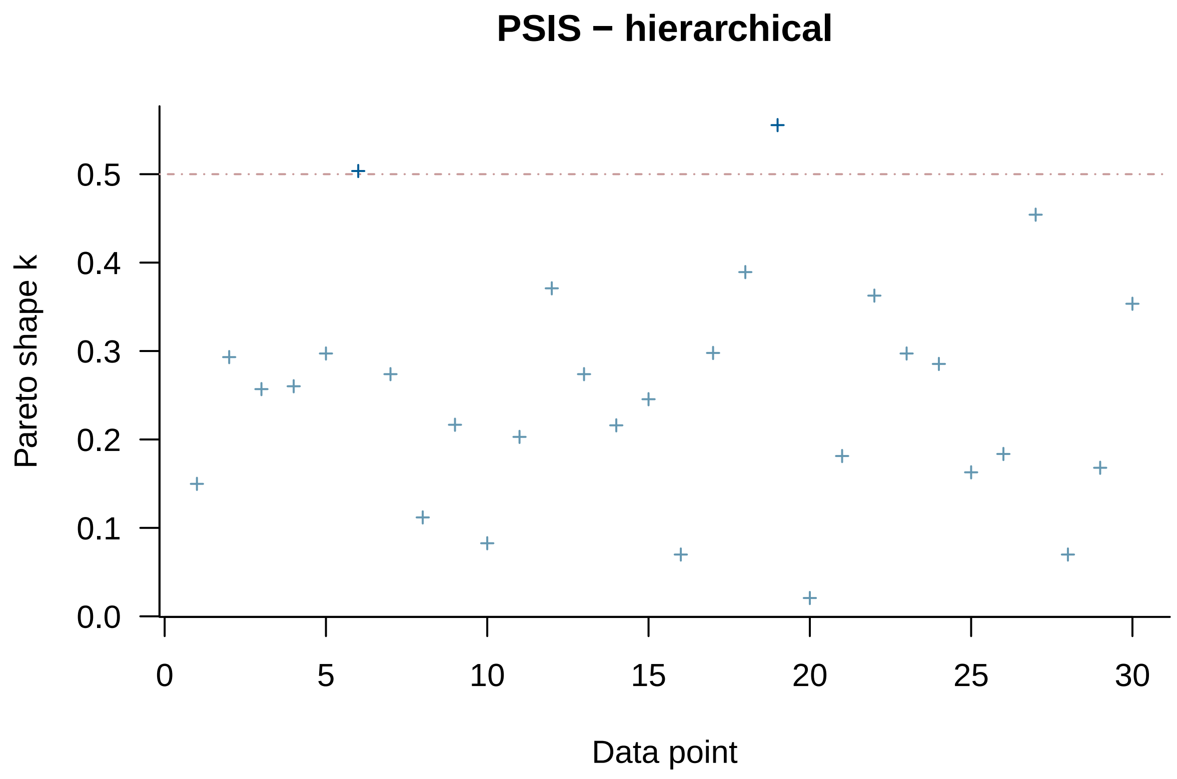
```
# hierarchial
log_lik_hierarchical <-
  extract_log_lik(fit_hierarchical, merge_chains = FALSE)
r_eff_hierarchical <-
  relative_eff(exp(log_lik_hierarchical))
loo_hierarchical <-
  loo(log_lik_hierarchical, r_eff = r_eff_hierarchical)
```

```
## Warning: Some Pareto k diagnostic values are slightly high. See help('pareto-k-diagnostic') for deta
```

```
print(loo_hierarchical)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -127.2 4.5
## p_loo         5.4 1.5
## looic       254.5 9.0
## ------
## 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%   682
##  (0.5, 0.7]   (ok)        2     6.7%   264
##    (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.
```

```r
# visualizing:
plot(loo_hierarchical, diagnostic = c("k", "n_eff"),
     label_points = FALSE, main = "PSIS - hierarchical")
```



For the hierachical model we get:

- PSIS-LOO : -127.2
- $\hat{k}$: All Pareto k estimates are ok (k $<$ 0.7)

Q3

The computation of the effective number of parameters p_eff for each of the three models was done in the previous question with the loo function. Printing the results again:

```r
print(loo_separate)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -132.6 3.2
## p_loo         9.9 1.1
## looic       265.2 6.3
## ------
## Monte Carlo SE of elpd_loo is NA.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     25    83.3%   442
##  (0.5, 0.7]   (ok)        2     6.7%   274
##    (0.7, 1]   (bad)       2     6.7%   77
##    (1, Inf)   (very bad)  1     3.3%   64
## See help('pareto-k-diagnostic') for details.
```

```r
print(loo_pooled)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -131.0 4.3
## p_loo         2.1 0.8
## looic       262.1 8.6
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     29    96.7%   1896
##  (0.5, 0.7]   (ok)        1     3.3%   510
##    (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.
```

```r
print(loo_hierarchical)
```

```
##
## Computed from 4000 by 30 log-likelihood matrix
##
```

```
##          Estimate  SE
## elpd_loo   -127.2 4.5
## p_loo         5.4 1.5
## looic       254.5 9.0
## ------
## 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%   682
##  (0.5, 0.7]   (ok)        2     6.7%   264
##    (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.
```

Q4

In Q2 we answered this question with computations and plots. Summarizing the results from Q2:

In the first plot (separated) we see three obersavations with pareto $\hat{k} > 0.7$, which indicates that the model are not reliable. However the two last two diagnostic plots only shows $\hat{k} < 0.7$, which means these models are reliable.

Q5

To compare the models to each other we can use the compare function;

```
compare(loo_separate, loo_pooled)
```

```
## elpd_diff        se
##       1.6       4.0
```

```
compare(loo_separate, loo_hierarchical)
```

```
## elpd_diff        se
##       5.4       3.1
```

```
compare(loo_pooled, loo_hierarchical)
```

```
## elpd_diff        se
##       3.8       1.7
```

We do see a diference between all the models. To summarize the results obtained in Q1-Q4:

Separate model

- PSIS-LOO : -132.6, the highest of the three models (the smaller the better)
- $\hat{k}$: 90% good or ok. 10% bad

Pooled model

- PSIS-LOO : -131.
- $\hat{k}$ 100 % good or ok

Hierarchical model

- PSIS-LOO : -127.2, smallest of the three models (and hereby best)
- $\hat{k}$: 100 % good or ok

Overall the most reliable model is the Hierachical.

References:

Based on code examples from:
https://github.com/avehtari/BDA_R_demos