Assignment 8

Anonymous

Contents

Load packages and import data

```
library(aaltobda)
library(cmdstanr)
library(loo)
library(bayesplot)
library(ggplot2)
library(posterior)

data("factory")
```

1. Fitting of the models

- Separate model

```
separate_m <- cmdstan_model(stan_file="separate_model.stan")
separate_m
```

```
data {
  int<lower=0> N; // number of observations
  int<lower=0> K; // number of groups
  array[N] int<lower=1, upper=K> g; // discrete group indicators
  vector<lower=0>[N] y; // observations
}

parameters {
  vector[K] mu; // group means
  vector<lower=0>[K] sigma; // group standard deviations
}

model {
  mu ~ normal(0, 10);
  sigma ~ gamma(1,1);
  y ~ normal(mu[g], sigma[g]);
}

generated quantities {
  real ypred;
  vector[N] log_lik;
  for (i in 1:N)
    log_lik[i] = normal_lpdf(y[i] | mu[g[i]], sigma[g[i]]);// predictive distribution for the sixth mac

  ypred = normal_rng(mu[6], sigma[6]);
}
```

```
separate_data <- list(
  N = 6 * nrow(factory),
  K = ncol(factory),
  g = rep(1:6, nrow(factory)),
  y = c(t(factory[,1:6]))
)
```

```
fit_separate <- separate_m$sample(data=separate_data, refresh=1000, seed=42)
```

```
Running MCMC with 4 sequential chains...

Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1 finished in 0.5 seconds.
Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2 finished in 0.4 seconds.
Chain 3 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 3 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 3 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 3 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 3 finished in 0.4 seconds.
Chain 4 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 4 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 4 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 4 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 4 finished in 0.4 seconds.

All 4 chains finished successfully.
Mean chain execution time: 0.4 seconds.
Total execution time: 2.1 seconds.
```

- Pooled model

```
pooled_m <- cmdstan_model(stan_file="pooled_model.stan")
pooled_m
```

```
data {
  int<lower=0> N; // number of observations
  vector<lower=0>[N] y; // observations
}

parameters {
  real mu;
  real<lower=0> sigma;
}

model {
  mu ~ normal(0, 10);
  sigma ~ gamma(1,1);
  y ~ normal(mu, sigma);
}

generated quantities {
  real ypred;
```

```
  vector[N] log_lik;
  for (i in 1:N)
    log_lik[i] = normal_lpdf(y[i] | mu, sigma); // predictive distribution for the sixth machine

  ypred = normal_rng(mu, sigma);
}
```

```
pooled_data <- list(
  N = 6 * nrow(factory),
  y = c(t(factory[,1:6]))
)

fit_pooled <- pooled_m$sample(data=pooled_data, refresh=1000, seed=42)
```

```
Running MCMC with 4 sequential chains...

Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1 finished in 0.1 seconds.
Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2 finished in 0.1 seconds.
Chain 3 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 3 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 3 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 3 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 3 finished in 0.1 seconds.
Chain 4 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 4 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 4 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 4 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 4 finished in 0.1 seconds.

All 4 chains finished successfully.
Mean chain execution time: 0.1 seconds.
Total execution time: 0.7 seconds.
```

- Hierarchical model

```
hierarchical_m <- cmdstan_model(stan_file="hierarchical_model.stan")
hierarchical_m
```

```
data {
  int<lower=0> N; // number of observations
  int<lower=0> K; // number of groups
  array[N] int<lower=1, upper=K> g; // discrete group indicators
  vector<lower=0>[N] y; // observations
}
```

```
parameters {
  real mu0;
  vector[K] mu;
  real<lower=0> sigma;
}

model {
  mu0 ~ normal(0, 10);
  sigma ~ gamma(1,1);
  mu ~ normal(mu0, sigma);
  y ~ normal(mu[g], sigma);
}

generated quantities {
  real ypred;
  vector[N] log_lik;
  for (i in 1:N)
    log_lik[i] = normal_lpdf(y[i] | mu[g[i]], sigma);// predictive distribution for the sixth machine

  ypred = normal_rng(mu[6], sigma);
}
```

```
hierarchical_data <- list(
  N = 6 * nrow(factory),
  K = ncol(factory),
  g = rep(1:6, nrow(factory)),
  y = c(t(factory[,1:6]))
)

fit_hierarchical <- hierarchical_m$sample(data=hierarchical_data, refresh=1000, seed=42)
```

```
Running MCMC with 4 sequential chains...

Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1 finished in 0.2 seconds.
Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 2 finished in 0.2 seconds.
Chain 3 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 3 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 3 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 3 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 3 finished in 0.2 seconds.
Chain 4 Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 4 Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 4 Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 4 Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 4 finished in 0.3 seconds.
```

```
All 4 chains finished successfully.
Mean chain execution time: 0.2 seconds.
Total execution time: 1.2 seconds.
```

2. Computation of the PSIS-LOO elpd values and the $\hat{k}$-values

- Separate model

```
separate_loo <- fit_separate$loo()
```

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

```
separate_loo
```

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

         Estimate   SE
elpd_loo   -196.8  8.0
p_loo        19.8  1.5
looic       393.6 15.9
------
Monte Carlo SE of elpd_loo is 0.1.

Pareto k diagnostic values:
                         Count Pct.    Min. n_eff
(-Inf, 0.5]   (good)     26    86.7%   538
 (0.5, 0.7]   (ok)        4    13.3%   461
   (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.
```

PSIS-LOO elpd is -197.

4 $\hat{k}$-values are between 0.5 and 0.7 and 26 values are lower than 0.5.

- Pooled model

```
pooled_loo <- fit_pooled$loo()
pooled_loo
```

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

         Estimate  SE
elpd_loo   -134.5 4.9
p_loo         2.5 0.8
looic       269.0 9.9
------
```

```
Monte Carlo SE of elpd_loo is 0.1.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

PSIS-LOO elpd is -135

All $\hat{k}$-values are lower than 0.5.

- Hierarchical model

```
hierarchical_loo <- fit_hierarchical$loo()
hierarchical_loo
```

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

         Estimate  SE
elpd_loo   -127.9 3.0
p_loo         5.5 1.0
looic       255.8 5.9
------
Monte Carlo SE of elpd_loo is 0.1.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

PSIS-LOO elpd is -128

All $\hat{k}$-values are lower than 0.5.

3. Computation of the effective number of parameters p_{eff}

- Separate model

```
separate_loo$estimates
```

```
           Estimate        SE
elpd_loo -196.78922  7.953282
p_loo      19.82498  1.466361
looic     393.57843 15.906563
```

p_{eff} for separate model is 19.8.

- Pooled model

```
pooled_loo$estimates
```

```
            Estimate        SE
elpd_loo -134.522760 4.9254797
p_loo       2.527918 0.8485453
looic     269.045520 9.8509593
```

p_{eff} for pooled model is 2.5.

- Hierarchical model

```
hierarchical_loo$estimates
```

```
           Estimate       SE
elpd_loo -127.904448 2.957054
p_loo       5.549513 1.026310
looic     255.808896 5.914107
```

p_{eff} for hierarchical model is 5.5.

4. Assessment of how reliable the PSIS-LOO estimates are for the three models based on the $\hat{k}$-values

All the PSIS-LOO estimates are reliable because all the $\hat{k}$-values are lower or equal than 0.7 in the presented models.

5. Model selection according to PSIS-LOO

The separate model has the worst $elpd_{loo\_cv}$ equal to -197.

Pooled and hierarchical models has almost equal estimations: 269 and 256 $elpd_{loo\_cv}$.

Let's compare these two models:

```
loo_compare(pooled_loo, hierarchical_loo)
```

```
       elpd_diff se_diff
model2  0.0       0.0
model1 -6.6       3.9
```

We got that the hierarchical model gave the better result, thus, it should be selected.