BDA - Assignment 8

Contents


```
library(aaltobda)
library(ggplot2)
library(StanHeaders)
library(rstan)
```

```
## 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.
```

```
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
```

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

1) Pooled Model

In the pooled model all the machines are considered as one entity, thus we have to combine the all the measurements into one and perform our prediction on the whole data, rather than a subset.
The model is defined as follows.

```
data {
int<lower=0> N; // number of data points
vector[N] y; //
}
parameters {
real mu; // group means
real<lower=0> sigma; // common std
}
```

```
model {
y ~ normal(mu, sigma);
}

generated quantities {
real ypred;
ypred = normal_rng(mu, sigma);
vector[N] log_lik;
for (i in 1:N)
log_lik[i] = normal_lpdf(y[i] | mu, sigma);
}
```

Then we need to fit it and use it to generate desired values.

```
pooled_factory_data<-as.vector(as.matrix(factory))
pooled_data<-list(N=length(pooled_factory_data),
                  y=pooled_factory_data)
pooled_fit<-stan(file='pooled_model_assignment8.stan',data=pooled_data)
```

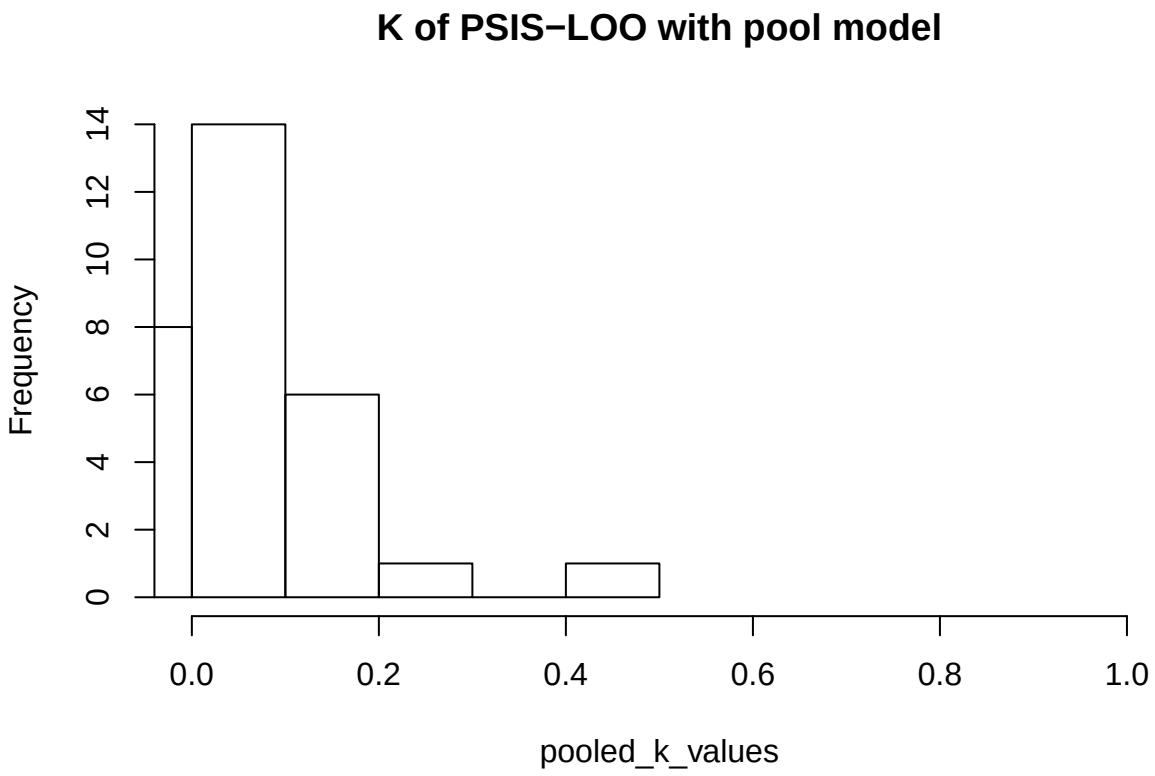
```
pooled_fit_ss <- extract(pooled_fit, permuted = TRUE)
pooled_log_lik <- pooled_fit_ss$log_lik
dims_pooled<-dim(pooled_log_lik)
S<-dims_pooled[1]
pooled_psis<-loo(pooled_log_lik)
pooled_elpd_loo<-pooled_psis$estimates[1,1]
pooled_p_loo<-pooled_psis$estimates[2,1]
cat("The PSIS-LOO value for pool model is:",pooled_elpd_loo,'\n' )
```

```
## The PSIS-LOO value for pool model is: -130.9185
```

```
cat("The p_ef value for pool model is:",pooled_p_loo ,'\n')
```

```
## The p_ef value for pool model is: 1.981066
```

```
pooled_k_values<-pooled_psis$diagnostics$pareto_k
hist(pooled_k_values,main ="K of PSIS-LOO with pool model",xlim=c(0.0,1.0))
```



2) Separate Model

In the separate model we treat every machine as an individual entity, thus when calculating σ or μ we take into consideration only a single machine. The combination of all machines should not effect σ or μ. The stan model is defined as follows.

```
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; // group stds
}
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]]);
```

```
}
```

Then we need to fit it and use it to generate desired values.

```
separate_data<-list(N=length(as.matrix(factory)),
                    K=6,
                    x=c(1, 1, 1, 1, 1,2, 2, 2, 2, 2,3, 3,
                        3, 3, 3,4, 4, 4, 4, 4,5, 5, 5, 5, 5,6, 6, 6, 6, 6),
                    y=as.vector(as.matrix(factory)))
separate_fit<-stan(file='separate_model_assignment8.stan',data=separate_data)
```

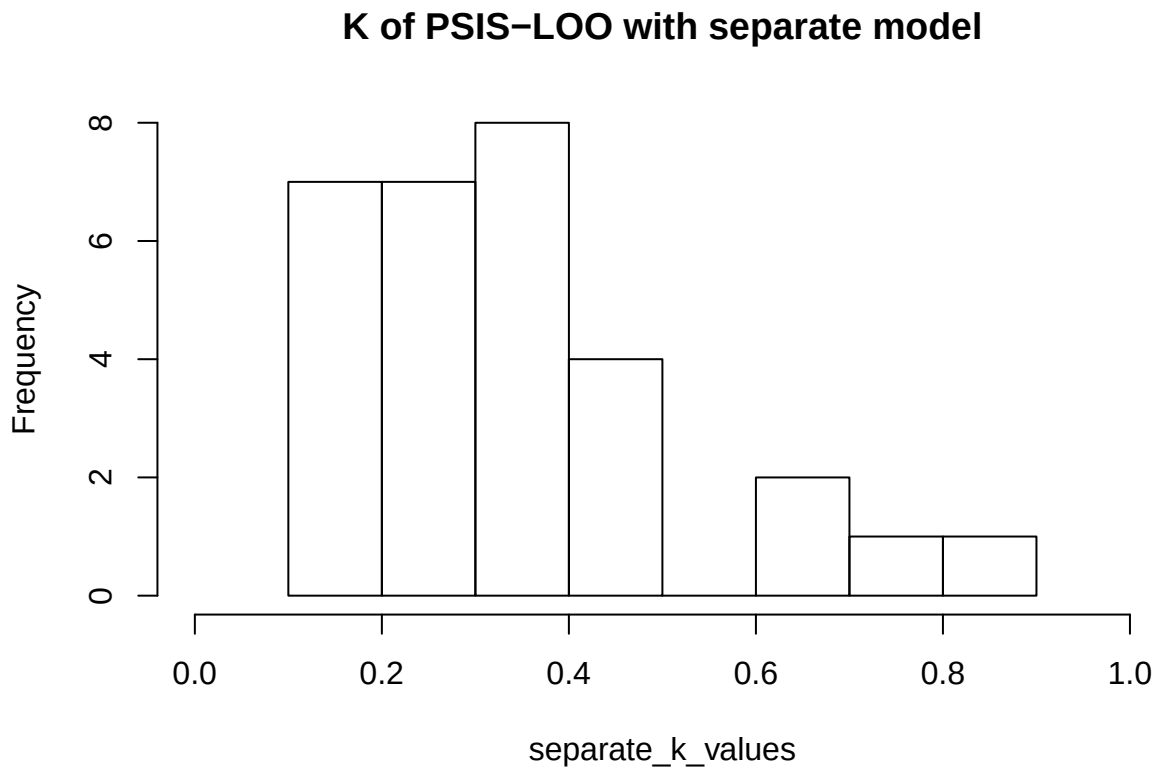
```
separate_fit_ss <- extract(separate_fit, permuted = TRUE)
separate_log_lik <- separate_fit_ss$log_lik
dims_separate<-dim(separate_log_lik)
S<-dims_separate[1]
separate_psis<-loo(separate_log_lik)
separate_elpd_loo<-separate_psis$estimates[1,1]
separate_p_loo<-separate_psis$estimates[2,1]
cat("The PSIS-LOO value for separate model is:",separate_elpd_loo,'\n' )
```

```
## The PSIS-LOO value for separate model is: -132.2948
```

```
cat("The p_ef value for separate model is:",separate_p_loo ,'\n')
```

```
## The p_ef value for separate model is: 9.616786
```

```
separate_k_values<-separate_psis$diagnostics$pareto_k
hist(separate_k_values,main ="K of PSIS-LOO with separate model",xlim=c(0.0,1.0))
```



```
separate_elpd_loo<-separate_psis$estimates[1,1]
separate_p_loo<-separate_psis$estimates[2,1]
```

3) Hierarchical Model

The hierarchical model is quite interesting in the sense that it can predict measurements for the machines which have no data. For example, there is no data about the seventh machine, but this model can predict its posterior distribution.
The stan model is defined as follows.

```
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 std
}
model {
mu ~ normal(mu0, sigma0);
y ~ normal(mu[x], sigma);
}
generated quantities {
real ypred6;
real mu7;
vector[N] log_lik;
ypred6 = 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);
}
```

Then we need to fit it and use it to generate desired values.

```
hierarchical_fit<-stan(file='hierarchical_model_assignment8.stan',data=separate_data)
```

```
## Warning: There were 10 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
```

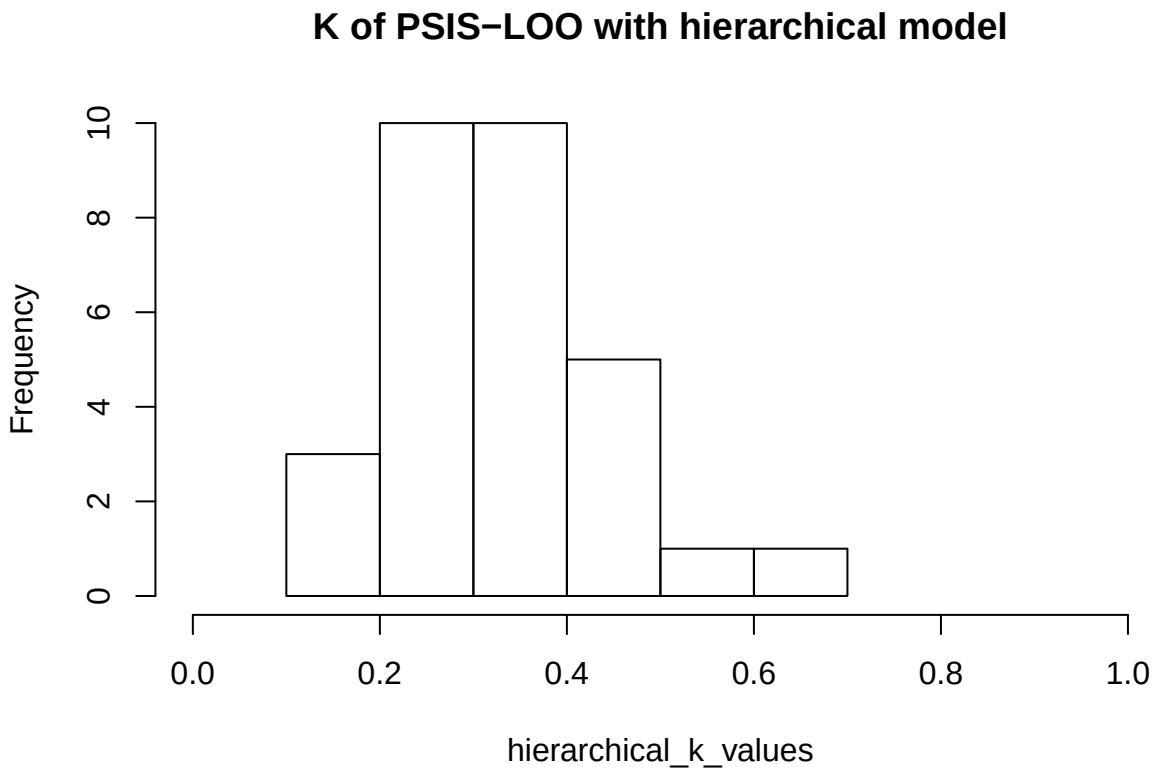
```
hierarchical_fit_ss <- extract(hierarchical_fit, permuted = TRUE)
hierarchical_log_lik <- hierarchical_fit_ss$log_lik
dims_hierarchical<-dim(hierarchical_log_lik)
S<-dims_hierarchical[1]
hierarchical_psis<-loo(hierarchical_log_lik)
hierarchical_elpd_loo<-hierarchical_psis$estimates[1,1]
hierarchical_p_loo<-hierarchical_psis$estimates[2,1]
cat("The PSIS-LOO value for hierarchical model is:",hierarchical_elpd_loo,'\n' )
```

```
## The PSIS-LOO value for hierarchical model is: -126.9759
```

```r
cat("The p_ef value for hierarchical model is:",hierarchical_p_loo ,'\n')
```

```
## The p_ef value for hierarchical model is: 5.737239
```

```r
hierarchical_k_values<-hierarchical_psis$diagnostics$pareto_k
hist(hierarchical_k_values,main ="K of PSIS-LOO with hierarchical model",xlim=c(0.0,1.0))
```



4) Summary

```r
cat("The PSIS-LOO value for pool model is:",pooled_elpd_loo,'\n' )
```

```
## The PSIS-LOO value for pool model is: -130.9185
```

```r
cat("The p_ef value for pool model is:",pooled_p_loo ,'\n')
```

```
## The p_ef value for pool model is: 1.981066
```

```r
cat("The PSIS-LOO value for separate model is:",separate_elpd_loo,'\n' )
```

```
## The PSIS-LOO value for separate model is: -132.2948
```

```r
cat("The p_ef value for separate model is:",separate_p_loo ,'\n')
```

```
## The p_ef value for separate model is: 9.616786
```

```r
cat("The PSIS-LOO value for hierarchical model is:",hierarchical_elpd_loo,'\n' )
```

```
## The PSIS-LOO value for hierarchical model is: -126.9759
```

```r
cat("The p_ef value for hierarchical model is:",hierarchical_p_loo ,'\n')
```

```
## The p_ef value for hierarchical model is: 5.737239
```

Based on the histograms of $\hat{k}$ values of these three models, we can see that the PSIS-LOO estimates of pool model is the most reliable one since its $\hat{k}$ values are all lower than 0.7. And the PSIS-LOO estimates of separate model is the least reliable one. For the PSIS-LOO estimates of hierarchical model, most of its $\hat{k}$ values are lower than 0.7 so it's totally acceptable.

Thus, according to PSIS-LOO, we should choose hierarchical model.