CS-E5710 Bayesian Data Analysis
Assignment 9

November 24, 2019

NB Source code is given in the Appendix.

1 Model

As noticed in the previous assignment, the hierarchical model is best with the dataset as it does treat every machine as a separate entity, but also computes the combination of all the machines as one entity. Hence the hierarchical model can predict quality of the machines even without data. In this case, there is no data about the seventh machine, but this model can predict its posterior distribution.

Stan code

```
stan_code_hierarchical = '''
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 {
    vector[K+1] ypred;
    real mu7;
    mu7 = normal_rng(mu0, sigma0);
    for (i in 1:K)
        ypred[i] = normal_rng(mu[i], sigma);
    ypred[K+1] = normal_rng(mu7, sigma);
}
'''
```

```
            mean se_mean     sd   2.5%    25%    50%    75%  97.5%  n_eff   Rhat
mu0        92.66     0.2   8.17  76.68  88.27   92.8  96.99 108.42   1625    1.0
sigma0     16.45    0.27   9.52    5.4  10.52  14.43  19.46  40.62   1269    1.0
mu[1]      79.61    0.16   6.65  66.76  75.25  79.56  83.87  92.76   1650    1.0
mu[2]      103.3    0.13   6.67  90.33  98.71 103.33 107.79 116.29   2792    1.0
mu[3]      88.97     0.1   6.12  76.63  85.11  88.89  92.92  101.3   3834    1.0
mu[4]     107.51    0.18   6.97  93.24 102.89 107.54 112.23 120.63   1565    1.0
mu[5]      90.51     0.1   6.24  78.09  86.54  90.53  94.44 103.06   3917    1.0
mu[6]      87.45    0.11   6.28  74.79  83.28  87.55  91.75  99.38   3174    1.0
sigma      15.19    0.06   2.38  11.35  13.47  14.94  16.63   20.3   1586    1.0
ypred[1]   79.61    0.27  16.52  47.85  68.65  79.07  90.56  113.7   3763    1.0
ypred[2]  102.92    0.27  16.68  68.54  92.04 103.06  113.8 134.52   3799    1.0
ypred[3]   89.32    0.27  16.37  56.74   78.5  89.45 100.23 121.98   3640    1.0
ypred[4]  107.71    0.28  16.93  74.23  96.62 107.66 118.62 141.03   3682    1.0
ypred[5]   90.74    0.27   16.8  57.82  79.67   90.6 101.54  124.7   3995    1.0
ypred[6]   87.27    0.28  16.66  55.05  76.18  87.31  98.43 120.51   3480    1.0
ypred[7]   92.64    0.42  25.84  41.86  76.79  92.81 108.35  145.2   3775    1.0
mu7        92.66    0.34  20.75  50.54  82.42  92.71 102.97 134.52   3697    1.0
lp__      -108.9    0.07   2.48 -114.8 -110.3 -108.5 -107.0 -105.2   1218    1.0
```

2 Compute the expected utilities

The predicted quality of products, ypred is used to compute the expected utilities.

```
utility = np.zeros(7)
ypred = fit_hierarchical.extract(permuted=True)['ypred']

for i in range(7):
    for j in range(0, len(ypred)):
        if ypred[j, i] < 85:
            utility[i] -= 106
        else:
            utility[i] += (200-106)

        i_utility = utility[i]/len(ypred)
```

Expected utilities

```
('Machine', 1, -33.85)
('Machine', 2, 67.4)
('Machine', 3, 16.5)
('Machine', 4, 76.6)
('Machine', 5, 20.9)
('Machine', 6, 5.25)
('Machine', 7, 19.4)
```

Ranked from worst to best

```
('Machine', 1, -33.85)
('Machine', 6, 5.25)
('Machine', 3, 16.5)
('Machine', 5, 20.9)
('Machine', 2, 67.4)
('Machine', 4, 76.6)
```

Expected utility for the 7th machine

```
('Machine', 7, 19.4)
```

Value for the expected utilities for the 1st machine is negative which means financial loss to the company, while the rest of the machines are expected to be profitable. The expected value for 7th machine is 19.4 which is expected to be profitable. Thus the company should buy a new (7th) machine.

A Code

```
import numpy as np
import pandas as pd
import pystan

machines = pd.read_fwf('./factory.txt', header=None).values
machines_transposed = machines.T

'''
Hierarchical model
'''
stan_code_hierarchical = '''
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 {
    vector[K+1] ypred;
    real mu7;
    mu7 = normal_rng(mu0, sigma0);
    for (i in 1:K)
        ypred[i] = normal_rng(mu[i], sigma);
    ypred[K+1] = normal_rng(mu7, sigma);
}
'''

model_hierarchical = pystan.StanModel(model_code=stan_code_hierarchical)
data_hierarchical = dict(
    N=machines_transposed.size,
    K=6,
    x=[
```

```
        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=machines_transposed.flatten()
)

fit_hierarchical = model_hierarchical.sampling(data=data_hierarchical, n_jobs=-1)
print(fit_hierarchical)

utility = np.zeros(7)
ypred = fit_hierarchical.extract(permuted=True)['ypred']
ulist=[]
for i in range(7):
    for j in range(0, len(ypred)):
        if ypred[j, i] < 85:
            utility[i] -= 106
        else:
            utility[i] += (200-106)

        i_utility = utility[i]/len(ypred)

    ulist.append(('Machine', i+1, i_utility))
    #print('Machine', i+1, i_utility)

for u in ulist:
    print(u)

sorted_ulist= sorted(ulist, key=lambda x: x[2])
for s in sorted_ulist:
    print(s)
```