Bayesian Statistics HW-5

Weijia Zhao

March 27, 2022

Caution: I keep 6 decimal digits for infinite decimals.

Exercise 1

Blood Volume in Infants
The data for 16 babies in whom the cord was clamped early the total blood are given by

$$D_1 = \{13.8, 8.0, 8.4, 8.8, 9.6, 9.8, 8.2, 8.0, 10.3, 8.5, 11.5, 8.2, 8.9, 9.4, 10.3, 12.6\}$$

The data for 16 babies in whom the cord was not clamped until the placenta began to descend are given by

$$D_2 = \{10.4, 13.1, 11.4, 9.0, 11.9, 16.2, 14.0, 8.2, 13.0, 8.8, 14.9, 12.2, 11.2, 13.9, 13.4, 11.9\}$$

Assume gamma likelihoods with different shape/rate parameters for two procedures and assume $Gamma(0.001, 0.001)$ prior distribution for both shape and rate.

For both procedures, we denote the data to be $\{x_i\}_{i=1}^n$ where $n = 16$. The likelihood is given by $x_i|(\alpha, \beta) \sim Gamma(\alpha, \beta)$ where $\alpha \sim Gamma(\alpha_1 = 0.001, \beta_1 = 0.001)$ and $\beta \sim Gamma(\alpha_2 = 0.001, \beta_2 = 0.001)$. Put everything in PyMC (2000 sample with 4 chains and burn in the first 1000 sample, note that the mean of Gamma distribution is given by α/β, thus $mudiff = \alpha_1/\beta_1 - \alpha_2/\beta_2$), we get the following results.

As we can see, the 95% (HDI) credible set is given by (−3.943, −0.949), does not contain 0

Table 1: Q1 Results

	mean	sd	hdi_2.5%	hdi_97.5%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
alpha1	35.145	12.744	12.882	61.065	0.1	0.073	16400.0	18274.0	1.0
beta1	3.644	1.33	1.304	6.332	0.01	0.008	16493.0	18567.0	1.0
alpha2	27.78	10.141	9.843	47.677	0.102	0.092	14154.0	8065.0	1.0
beta2	2.297	0.846	0.791	3.944	0.008	0.008	14131.0	8002.0	1.0
mudiff	-2.462	0.753	-3.943	-0.949	0.003	0.002	79681.0	61062.0	1.0

Exercise 2

Can Skull Variations in *Canis lupus L* Predict Habitat?
We do the following regression (Note that a constant term is added, following the standard practice)

$$P(Location = Arctic) = Logistics(\beta_0 + \beta_1 \cdot gender + \beta_2 \cdot x3 + \beta_3 \cdot x7)$$
$$Logistics(y) = \frac{e^y}{1 + e^y}$$

The prior distribution of β is assumed to be $N(0, 10^2)$ (independent) Again 4 chains with 20000 samples and 1000 burn in are used, the regression results are shown below.

Table 2: Q2 Regression Results

	mean	sd	hdi_2.5%	hdi_97.5%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
beta[0]	-3.666	7.773	-18.806	11.693	0.044	0.031	31713.0	38709.0	1.0
beta[1]	1.015	1.112	-1.128	3.205	0.006	0.004	35918.0	39650.0	1.0
beta[2]	4.507	2.325	-0.059	9.03	0.015	0.011	24459.0	32339.0	1.0
beta[3]	-11.387	5.412	-21.972	-0.884	0.032	0.023	29196.0	35365.0	1.0

To make predictions, we sample regression coefficients from the posterior and calculate the corresponding $P(Location = Arctic)$ based on the formula above as well as the new record $gender = 1, x_3 = 5.28, x_7 = 1.78$. For mean value we can take two approach that are equivalent asymptotically by LLN (by necessarily identical with finite sample), first is to directly take the mean of those resulted $P(Location = Arctic)$, second is to sample from Bernoulli distribution for actual outcome for each resulted $P(Location = Arctic)$ and then take average for these actual outcomes. The first approach yield more meaningful confidence set along the way (outcome in the second approach is discrete 0,1 so the credible set for actual outcome will only be [0,0] or [1,1] or [0,1] and the last one is the most likely one for a 95% credible set). In addition, the second approach introduce one more layer of randomness and thus sampling error.

The prediction results is 0.6857915865803711 using the first approach (and 0.667 using the second approach they are really close, especially consider the small sample we have) (note PyMC recommend me not to sample over 25 observations and thus the results might be a bit volatile), the credible set for this likelihood is (0.3445759929714419, 0.923494935424825), the credible set for the actual outcome is (0,1)

Exercise 3

Micronuclei

The data provided is in tabular form and we first translate it into a table with three columns: the first column is constant, the second one is dose and the third one is micronuclei. (eg: the value 976 in upleft most cell means that we will have 976 rows with (dose=1,micronuclei=0) in our translated table).

We run the following regression

$$E[Micronuclei|Dose] = Poisson(\lambda = e^{\beta_0+\beta_1 \cdot Dose})$$

The prior distribution of β is assumed to be $N(0, 10^6)$ (independent) 4 chains with 3000 samples and 500 burn in are used, the regression results are shown below.

Table 3: Q3 Regression Results

	mean	sd	hdi_2.5%	hdi_97.5%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
intercept	-2.815	0.065	-2.94	-2.691	0.001	0.001	2282.0	2606.0	1.0
coef_dose	0.671	0.02	0.633	0.71	0.0	0.0	2308.0	3000.0	1.0

The prediction is given by 0.626 (process exactly the same as Q2) This value seems to be

Table 4: Q3 Prediction Results

	mean	sd	hdi_2.5%	hdi_97.5%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
micro	0.626	0.792	0.0	2.0	0.001	0.001	299854.0	298869.0	1.0

reasonable as it is in between the sample mean for actual data when $Dose = 3$ (sample mean is 0.556) and $Dose = 4$ (sample mean is 0.778)

Q1 Code

```
import numpy as np
import matplotlib.pyplot as plt
import pymc3 as pm
import arviz as az


#data
if __name__=='__main__':
    #sample
    data_clamp=[13.8,8.0,8.4,8.8,9.6,9.8,8.2,
                8.0,10.3,8.5,11.5,8.2,8.9,9.4,10.3,12.6]
    data_unclamp=[10.4,13.1,11.4,9.0,11.9,16.2,
                  14.0,8.2,13.0,8.8,14.9,12.2,11.2,13.9,13.4,11.9]
    np.average(data_clamp)-np.average(data_unclamp)
    #prior parameters
    shape=0.001
    rate=0.001
    with pm.Model() as m:
        #priors
        alpha1=pm.Gamma('alpha1',alpha=shape,beta=rate)
        beta1=pm.Gamma('beta1',alpha=shape,beta=rate)
        alpha2 = pm.Gamma('alpha2', alpha=shape, beta=rate)
        beta2 = pm.Gamma('beta2', alpha=shape, beta=rate)
        mudiff = pm.Deterministic('mudiff',alpha1/beta1-alpha2/beta2)
        #likelihood
        y1=pm.Gamma('y1',alpha=alpha1,beta=beta1,observed=data_clamp)
        y2=pm.Gamma('y2',alpha=alpha2,beta=beta2,observed=data_unclamp)
        #start sampling
        trace=pm.sample(20000,chains=4,tune=1000,
                        init="jitter+adapt_diag",
                        random_seed=1,return_inferencedata=True,)
        #print results
        print(az.summary(trace,hdi_prob=0.95))
        az.summary(trace, hdi_prob=0.95).to_csv('q1.csv')
```

Q2 Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import pymc3 as pm
import arviz as az
import statsmodels.api as sm
import random
random.seed(12345)

if __name__=='__main__':
    data=pd.read_csv('q2_data.csv',header=None)
    header=['location','gender','m1','m2',
            'm3','m4','m5','m6','m7','m8','m9']
    data.set_axis(header, axis=1, inplace=True)
    data['const']=1

    #only keep the following variables
    #y(location), x(gender,x3,x7)
    y=data['location']
    x=data[['gender','m3','m7']].values
    xc=data[['const','gender','m3','m7']].values

    #sample for prediction
    x_pred=[1,5.28,1.78]
    xc_pred=[1,1,5.28,1.78]

    #frequentist results---with constant term
    f_model1=sm.Logit(y,xc).fit()
    print(f_model1.summary())
    print(f_model1.predict(xc_pred))
    # bayesian regression---with constant term
    with pm.Model() as m:
        xc_data=pm.Data('x',xc)
        y_data=pm.Data('y',y)
        #priors
        beta=pm.Normal('beta',mu=0,sigma=10,shape=xc.shape[1])
        #likelihood
        likelihood=pm.invlogit(beta[0]*xc_data[:,0]+
                               beta[1]*xc_data[:,1]
                               +beta[2]*xc_data[:,2]+
                               beta[3]*xc_data[:,3])
        location=pm.Bernoulli(name='location',p=likelihood,observed=y_dat
        trace=pm.sample(draws=20000,chains=4,tune=1000,
                        init="jitter+adapt_diag",
                        random_seed=4,target_accept=0.95,)
    print(az.summary(trace,hdi_prob=0.95))
    az.summary(trace, hdi_prob=0.95).to_csv('q2_reg.csv')
```

```
#prediction
new_obs=np.array([xc_pred])
pm.set_data({'x':new_obs},model=m)
ppc=pm.sample_posterior_predictive(trace,model=m,samples=50)
print(az.summary(ppc,hdi_prob=0.95))
#manual prediction with likelihood
select_index=np.random.randint(0,trace['beta'].shape[0],500)
resulted_likelihood=[]
def cal_like(coeff,input_v):
    p=np.dot(coeff,input_v)
    return np.exp(p)/(1+np.exp(p))
for i in range(len(select_index)):
    resulted_likelihood.append(cal_like(trace['beta'][i], xc_pred))
print('mean, 2.5, 97.5 are '+
      str(np.average(resulted_likelihood))+
      ', '+str(np.percentile(resulted_likelihood,2.5))+
      ', '+str(np.percentile(resulted_likelihood,97.5))+
      'respectively')


print('finished')
```

Q3 Code

```
import numpy as np
import matplotlib.pyplot as plt
import pymc3 as pm
import arviz as az
import pandas as pd

if __name__=='__main__':
    data_raw = pd.read_csv('q3_data.csv', header=None).values
    doses=[0,0.5,1,2,3,4]
    micronucleis=[0,1,2,3,4,5,6]

    #expand the dataset to two variable
    data=[]
    for i in range(len(doses)):
        for j in range(len(micronucleis)):
            dose=doses[i]
            micronuclei=micronucleis[j]
            for k in range(data_raw[i][j]):
                data.append((dose,micronuclei))
    data = pd.DataFrame(data, columns=['dose','micronuclei'])
    data['const']=1

    x=data[['const','dose']].values
    y=data['micronuclei'].values
    with pm.Model() as m:
        #associate data with model
        x_data=pm.Data('x',x)
        y_data=pm.Data('y',y)
        #priors
        b0=pm.Normal('intercept',mu=0,sigma=10e3)
        b1=pm.Normal('coef_dose',mu=0,sigma=10e3)
        micro=pm.Poisson('micro',
                         mu=np.exp(b0*x_data[:,0]+
                                   b1*x_data[:,1]),
                         observed=y_data)
        trace = pm.sample(draws=3000, chains=4,tune=500,
                          init="jitter+adapt_diag",
                          random_seed=4,target_accept=0.95, )
    print(az.summary(trace, hdi_prob=0.95))
    az.summary(trace, hdi_prob=0.95).to_csv('q3.csv')

    #prediction
    with m:
        x_pred = [1,3.5]
        new_obs = np.array([x_pred])
        pm.set_data({'x': new_obs})
        ppc = pm.sample_posterior_predictive(trace, samples=50)
```

```
        print(az.summary(ppc, hdi_prob=0.95))
        az.summary(ppc, hdi_prob=0.95).to_csv('q3_pred.csv')

print('finished')
```