cs146-problem-set-1

September 29, 2023

1 CS146 Problem Set 1: Automated Inference with PyMC

- Complete all the problems below.
- Show and explain your work in detail.
- **Edit and submit this Jupyter notebook.** Do not upload only a PDF. Your instructor needs the notebook to run your code and check that it correctly reproduces your results.
- **Do not use ChatGPT** or similar AI tools in this assignment.
- **Learning outcomes:** #CS-ComputationalTools, #dataviz, #distributions, #optimization, #professionalism.

1.1 Problem 1. Fit a posterior

Tasks:

- Use PyMC to fit the model below to the data set provided below.
- Check that the sampler is working correctly.
- Plot and interpret the posterior.
- Show all your work in detail.

Model:

$$p \sim \text{Beta}(\alpha = 2, \beta = 2)$$

$$x_i \sim \text{Negative-Binomial}(r = 3, p) \quad i = 1, \ldots, 20$$

Defining the Model

The first step is to define the model. We need the prior and likelihood which are denoted above.

The prior defines our beliefs about the parameters before seeing the data. This could be obtained from historical information about the data generation process or what we believe to be true about the data before seeing it.

For our model the prior is defined as:

$$p \sim \text{Beta}(\alpha = 2, \beta = 2)$$

This prior with

$$\alpha = 2, and, \beta = 2)$$

tells us that we believe it's equally likely for

$$p$$

to be less than 0.5 or greater than 0.5. This prior suggests a balanced expectation regarding the value of

$$p$$

Both alpha and beta being > 1 means that the values are pulled away from the extreme ends of 0 or 1. This is still a relatively flat distribution which indicates that we are generally uncertain about the true value of p.

The likelihood on the other hand is a function that quantifies how well our model explains observed data. The likelihood is paired with the prior to compute the posterior distribution of the model parameters given the data. The likelihood, updates our prior beliefs about the model parameters based on the data observed:

$$Posterior \sim Likelihood \times Prior$$

For our model, the likelihood function is:

$$x_i \sim \text{Negative-Binomial}(r = 3, p) \quad i = 1, \ldots, 20$$

p (the parameter being modeled) is assumed to come from a Beta distribution which we specified in the prior. The likelihood function is conditioned on observed data

$$xi$$

and quantifies how probable the observed data is under different values of parameter

$$p$$

Beta distribution is also a conjugate prior for the binomial and negative binomial likelihoods. This means that the posterior distribution will also be a Beta distribution.

Given this, we can now use PyMC to fit the model above to the data set provided below.

[13]:
```
# Importing necessary libraries
import pymc as pm
import arviz as az
import numpy as np
import scipy.stats as sts
import matplotlib.pyplot as plt
```

[14]:
```
data = [9, 15, 4, 7, 7, 1, 3, 5, 8, 7, 9, 8, 4, 3, 0, 4, 6, 9, 2, 4]
```

[15]:
```
# Defining the model
with pm.Model() as data_model:
    p = pm.Beta('p', alpha = 2, beta = 2) # prior
```

```
    x = pm.NegativeBinomial('x', mu = 3/(1- p), alpha = 3, observed = data) #␣
↪likelihood
```

Sampling

We're interested in the posterior distribution of the parameters given the data. However, directly computing the posterior can be analytically heavy, especially in complex models. We can however draw samples from the posterior distribution, which provides an empirical representation of the posterior.

[16]:
```
# Step 2: Sampling from the model to get the posterior
with data_model:
    inference = pm.sample(cores = 1)
```

```
Auto-assigning NUTS sampler…
Initializing NUTS using jitter+adapt_diag…
Sequential sampling (2 chains in 1 job)
NUTS: [p]
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
Sampling 2 chains for 1_000 tune and 1_000 draw iterations (2_000 + 2_000 draws
total) took 2 seconds.
We recommend running at least 4 chains for robust computation of convergence
diagnostics
```

Checking the Sampler

In order to check wheher the sampler is working correctly, we can use a trace plot, a rank plot and check the summary of the posterior.

Trace Plot

This displays the values of a given parameter across the iterations of one or more MCMC (Markov chain Monte Carlo) chains. Each chain is plotted as a separate line.

Rank Plot

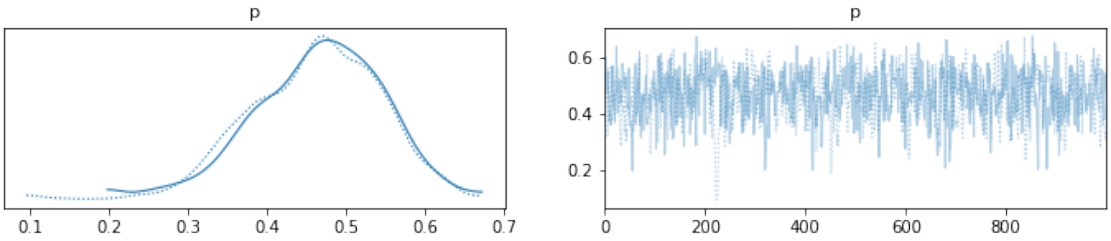
This displays the rank of the sampled values across different chains. The values are ranked within each chain, and then the ranks are plotted against the cumulative distribution function (CDF) of a uniform distribution.

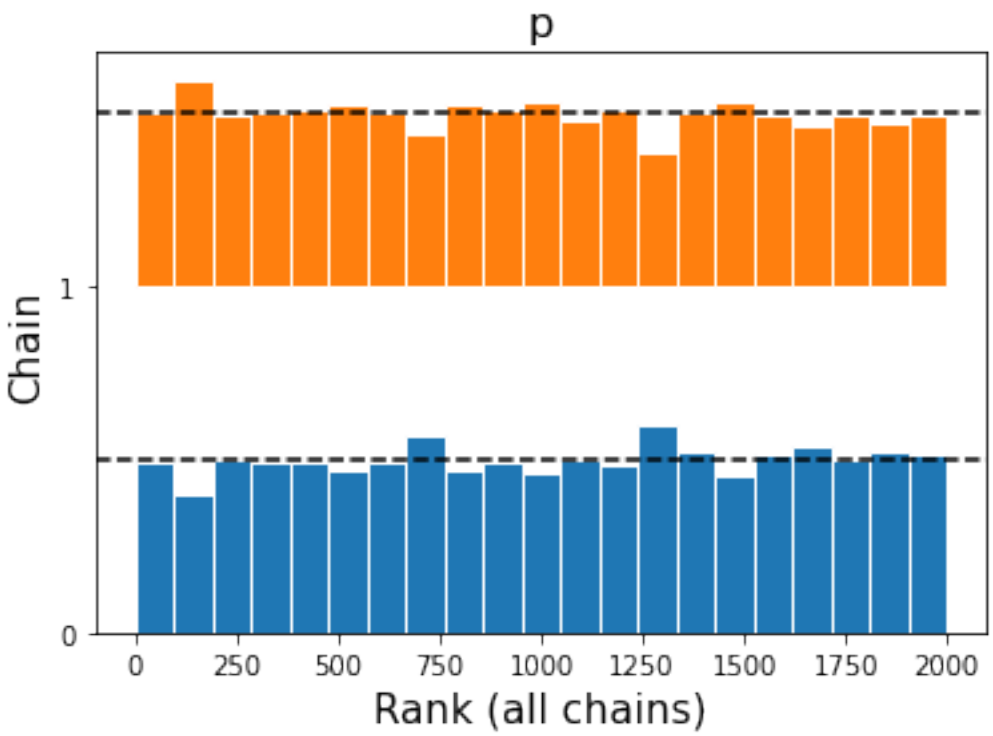
[19]:
```
az.plot_trace(inference)
az.plot_rank(inference)
az.summary(inference)
```

[19]:

	mean	sd	hdi_3%	hdi_97%	mcse_mean	mcse_sd	ess_bulk	ess_tail	\
p	0.463	0.083	0.321	0.619	0.003	0.002	665.0	808.0	

```
   r_hat
p    1.0
```





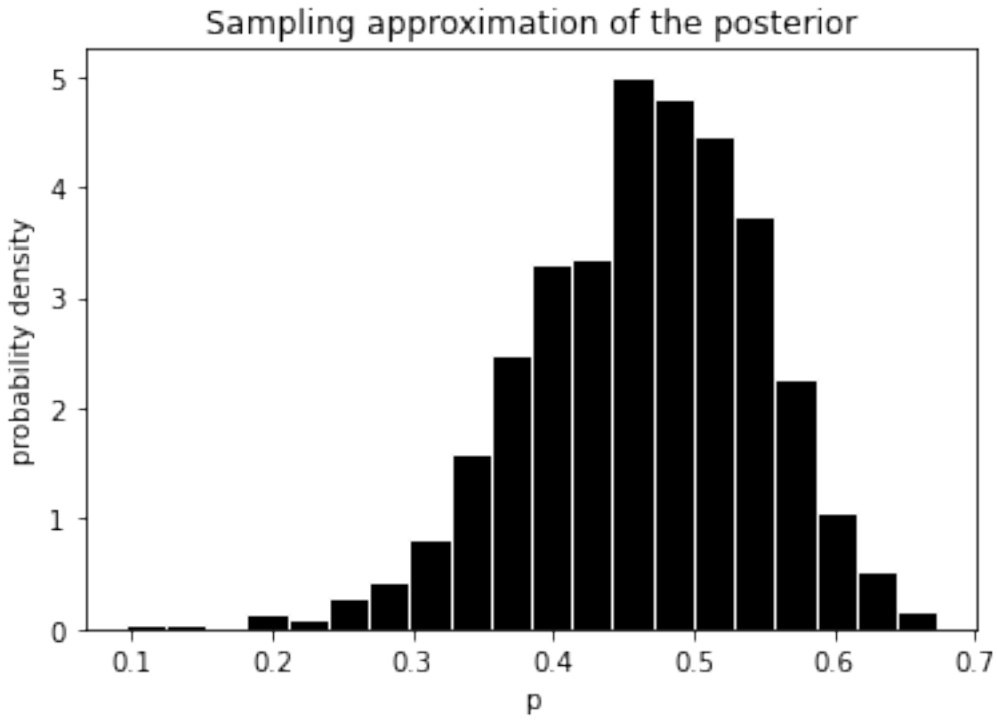
From the plots above, the sampler seems to be working well. This is because, in the trace plot, both chains are converging to the same distribution and overlap each other nicely which indicates that they covered the values uniformly. The chains are also well mixed and aren't stuck in any particular region. From the rank plot, the distribution is fairly uniform. This supports the conclusion drawn from the trace plot that the sampler is indeed working correctly.

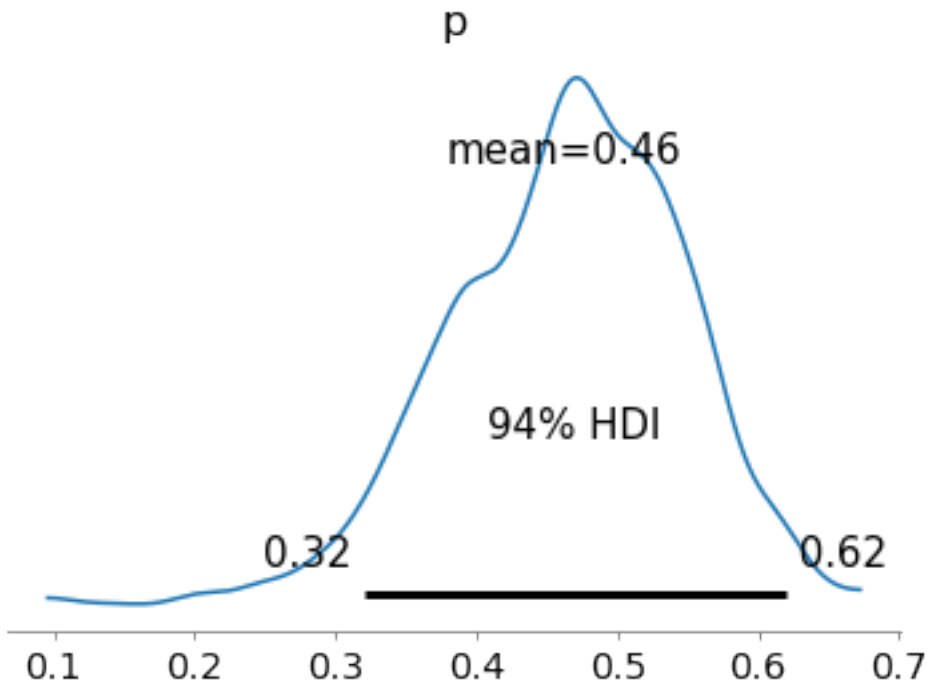
Plotting the Posterior

[33]:
```
all_p_samples = inference.posterior.p.values.flatten() # extract samples

# Plot the samples from the approximate posterior
plt.title('Sampling approximation of the posterior')
plt.xlabel('p')
plt.ylabel('probability density')
plt.hist(
    all_p_samples, bins=20, density=True,
    edgecolor='white', color = 'black')
plt.show()

# plotting posterior from arviz
az.plot_posterior(inference)
plt.show()
```





The mean of the posterior distribution for p is 0.46. This represents the expected value of p given the observed data and the model. In practical terms, this is our best point estimate for p based on the current data and model.

The 94% HDI represents an interval estimate of p. This interval contains 94% of the posterior probability, and is the most credible range of values for p given the data and the model. The interval [0.32, 0.62] is fairly wide, which reflects a substantial amount of uncertainty regarding the true value of p. The interval does not include 0 or 1, which suggests that, it's quite unlikely that p is at either extreme.

1.2 Problem 2. Create a prior

We want to model purchasing behavior at an e-commerce company. For each customer, we model the average amount of money they spend per month. Set up an appropriate model (likelihood and prior) such that the median of the average amount of money spent per customer per month is JPY 10,000 (ten thousand Japanese yen) and that 80% of customers have an average spend between JPY 1,000 and JPY 30,000.

- Motivate your choices for the prior distribution and likelihood function.
- Show that samples from your prior-predictive distribution have the percentiles indicated above. (This can be approximate but you have to quantify how close you are to the desired values.)

Choosing Prior and Likelihood

For the likelihood function, I chose a Gamma Distribution which has parameters

$$\alpha, and, \beta$$

This is denoted as:

$$X \sim Gamma(\alpha, \beta)$$

The Gamma distribution is flexible when modelling positive continuous values which applies to the context of our scenario since the value is money. The support of Gamma distribution is 0 to infinity which mimics spending habits as well. Since spending habits can take on any form of shape, Gamma is a good choice because we can tailor the parameters to match the given percentiles.

For the prior, we can pick the conjugate prior to the Gamma distribution which is the inverse Gamma distribution and finetune the parameters as well to match what we know.

These are denoted as:

$$\alpha_h \sim InverseGamma(\alpha, \beta)$$

$$\beta_h \sim InverseGamma(\alpha, \beta)$$

[203]:
```
# Generating data to feed into the model.
# Set parameters to match info given
shape = 2.5  # alpha
scale = 5000  # theta, so that the mean (shape*scale) is 10000

# Generate synthetic data
np.random.seed(0)  # for reproducibility
data2 = np.random.gamma(shape, scale, size=1000)  # generating 1000 data points
```

[204]:
```
# Defining the model
with pm.Model() as model:
    # Inverse Gamma Priors
    alpha = pm.InverseGamma('alpha', alpha=2, beta=6.5)
    beta = pm.InverseGamma('beta', alpha=5400, beta=2)

    # Gamma likelihood
    pm.Gamma('spending', alpha=alpha, beta=beta, observed=data)
```

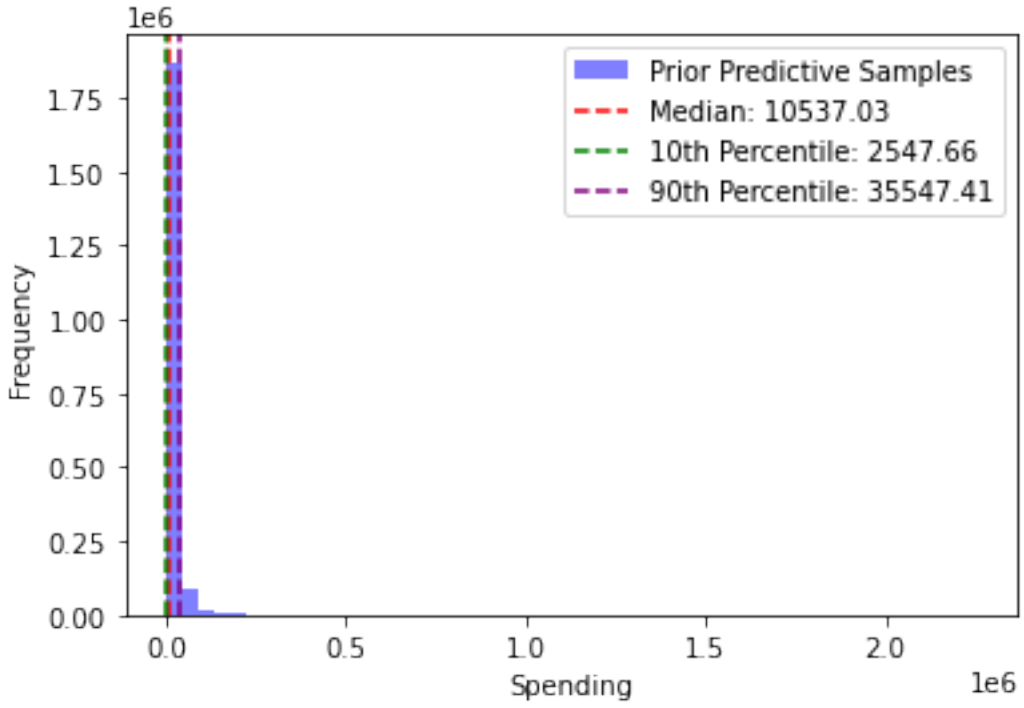
[205]:
```
# Draving samples from the prior-predictive distribution
n_samples = 2000
with model:
    inference = pm.sample_prior_predictive(n_samples)
inference
```

```
Sampling: [alpha, beta, spending]
```

[205]:
```
Inference data with groups:
        > prior
        > prior_predictive
        > observed_data
```

[206]:
```
# Plotting th prior predictive samples and percentiles
spending_samples = inference.prior_predictive.spending.values.flatten()

plt.figure()
plt.hist(spending_samples, bins=50, alpha=0.5, color='blue', label='Prior␣
 ↪Predictive Samples')
plt.axvline(x=np.median(spending_samples), color='red', linestyle='--',␣
 ↪label=f'Median: {np.median(spending_samples):.2f}')
plt.axvline(x=np.percentile(spending_samples, 10), color='green',␣
 ↪linestyle='--', label=f'10th Percentile: {np.percentile(spending_samples,␣
 ↪10):.2f}')
plt.axvline(x=np.percentile(spending_samples, 90), color='purple',␣
 ↪linestyle='--', label=f'90th Percentile: {np.percentile(spending_samples,␣
 ↪90):.2f}')
plt.xlabel('Spending')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



The prior-predictive distribution is the distribution obtained from observations that are expected before observing any data. By sampling from the prior predictive distribution, we aim to check if the model setup comprising of both the chosen priors and likelihood - aligns with the specified scenario before observing any actual data.

Upon analyzing the generated prior predictive samples, we observe that the median spending amount is JPY 10,537.03, the 10th percentile is JPY 2,547.66, and the 90th percentile is JPY 35,547.41. These values match the expectations set forth in the task scenario, i.e., a median of JPY 10,000 and 80% of customers having an average spend between JPY 1,000 and JPY 30,000. Although the obtained median and percentiles are not exact, they are reasonably close to the desired values, indicating that the model setup is a good starting point for understanding the purchasing behavior at the e-commerce company.

AI Use: Just Googled information on the possible distribution for the second question and Google AI gave me the idea for Gamma and I looked up the properties and conjugate prior on Wikipedia.

https://en.wikipedia.org/wiki/Gamma_distribution

[ ]: