Assessed Coursework 4

CID 01252821

Question 1

Part a

Part i

The marginal likelihood is given by

$$p(x) = \int_0^\infty \int_{-\infty}^\infty p(x|\mu, \sigma^{-2}) p(\mu, \sigma^{-2}) d\mu d\sigma^{-2}$$

The likelihood under the normal distribution is

$$p(x|\mu, \sigma) = (2\pi\sigma^2)^{-\frac{n}{2}} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^n (x_i-\mu)^2}.$$

The joint distribution is

$$p(\mu, \sigma^{-2}) = p(\mu|\sigma^{-2}) p(\sigma^{-2}),$$

where

$$p(\mu|\sigma^{-2}) = \sqrt{\lambda}(2\pi\sigma^2)^{-\frac{1}{2}} e^{-\frac{\lambda}{2\sigma^2}\mu^2},$$

$$p(\sigma^{-2}) = \frac{b^a}{\Gamma(a)} \sigma^{-2(a-1)} e^{-\frac{b}{\sigma^2}},$$

leading to

$$p(x) = \int_0^\infty \int_{-\infty}^\infty \left[(2\pi\sigma^2)^{-\frac{n}{2}} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^n (x_i-\mu)^2}\right] \left[\sqrt{\lambda}(2\pi\sigma^2)^{-\frac{1}{2}} e^{-\frac{\lambda}{2\sigma^2}\mu^2}\right] \left[\frac{b^a}{\Gamma(a)} \sigma^{-2(a-1)} e^{-\frac{b}{\sigma^2}}\right] d\mu d\sigma^{-2}.$$

Let's explore the following term

$$\sum_{i=1}^n (x_i - \mu)^2 = \sum_{i=1}^n \left[(x_i - \bar{x}) - (\mu - \bar{x})\right]^2 =$$

$$= \sum_{i=1}^n (x_i - \bar{x})^2 + \sum_{i=1}^n (\bar{x} - \mu)^2 - 2\sum_{i=1}^n (x_i - \bar{x})(\mu - \bar{x}) =$$

$$= \sum_{i=1}^n (x_i - \bar{x})^2 + \sum_{i=1}^n (\bar{x} - \mu)^2 = \sum_{i=1}^n (x_i - \bar{x})^2 + n(\bar{x} - \mu)^2,$$

where

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

$$p(x) = \int_0^\infty \int_{-\infty}^\infty \left[(2\pi\sigma^2)^{-\frac{n}{2}} e^{-\frac{1}{2\sigma^2}(\sum_{i=1}^n (x_i-\bar{x})^2 + n(\bar{x}-\mu)^2)}\right] \left[\sqrt{\lambda}(2\pi\sigma^2)^{-\frac{1}{2}} e^{-\frac{\lambda}{2\sigma^2}\mu^2}\right] \left[\frac{b^a}{\Gamma(a)}\sigma^{-2(a-1)} e^{-\frac{b}{\sigma^2}}\right] d\mu d\sigma^{-2}.$$

Now, let's observe

$$\lambda\mu^2 + n(\bar{x}-\mu)^2 = \lambda\mu^2 + \frac{\lambda+n}{\lambda+n} n(\bar{x}-\mu)^2 =$$

$$= (\lambda+n)\mu^2 - 2\mu n\bar{x} + \frac{n^2}{\lambda+n}\bar{x}^2 + \frac{\lambda n}{\lambda+n}\bar{x}^2 =$$

$$= (\lambda+n)\left(\mu - \frac{n\bar{x}}{\lambda+n}\right)^2 + \frac{\lambda n \bar{x}^2}{\lambda+n}.$$

Therefore, the marginal likelihood can be written as

$$p(x) = \int_0^\infty \sqrt{\lambda}(2\pi\sigma^2)^{-n/2}(2\pi\sigma^2)^{-1/2} e^{-\frac{\sigma^{-2}}{2}\sum_{i=1}^n (x_i-\bar{x})^2} e^{-\frac{\sigma^{-2}}{2}\frac{\lambda n\bar{x}^2}{\lambda+n}} \frac{b^a}{\Gamma(a)}\sigma^{-2(a-1)} e^{-b\sigma^{-2}} \int_{-\infty}^\infty e^{-\frac{\sigma^{-2}}{2}(\lambda+n)(\mu-\frac{n\bar{x}}{\lambda+n})^2} d\mu d\sigma^{-2} =$$

$$= (2\pi)^{-n/2}\frac{\sqrt{\lambda}}{(\lambda+n)^{1/2}}\frac{b^a}{\Gamma(a)}\int_0^\infty \sigma^{-2(a+\frac{n}{2}-1)} e^{-\frac{\sigma^{-2}}{2}\sum_{i=1}^n (x_i-\bar{x})^2} e^{-\frac{\sigma^{-2}}{2}\frac{\lambda n\bar{x}^2}{\lambda+n}} e^{-b\sigma^{-2}} d\sigma^{-2} =$$

$$= (2\pi)^{-n/2}\frac{\sqrt{\lambda}}{(\lambda+n)^{1/2}}\frac{b^a}{\Gamma(a)}\int_0^\infty \sigma^{-2(a+\frac{n}{2}-1)} e^{-\frac{\sigma^{-2}}{2}\sum_{i=1}^n (x_i-\bar{x})^2 - \frac{\sigma^{-2}}{2}\frac{\lambda n\bar{x}^2}{\lambda+n} - b\sigma^{-2}} d\sigma^{-2} =$$

$$= (2\pi)^{-n/2}\frac{\sqrt{\lambda}}{(\lambda+n)^{1/2}}\frac{b^a}{\Gamma(a)}\int_0^\infty \sigma^{-2(a+\frac{n}{2}-1)} e^{-\frac{\sigma^{-2}}{2}\left(\sum_{i=1}^n (x_i-\bar{x})^2 + \frac{\lambda n\bar{x}^2}{\lambda+n} + 2b\right)} d\sigma^{-2}.$$

Let's simplify

$$\frac{1}{2}\left(\sum_{i=1}^n (x_i-\bar{x})^2 + \frac{\lambda n\bar{x}^2}{\lambda+n} + 2b\right) = \frac{1}{2}\sum_{i=1}^n x_i^2 - \sum_{i=1}^n x_i\bar{x} + \frac{1}{2}\sum_{i=1}^n \bar{x}^2 + \frac{1}{2}\frac{\lambda n\bar{x}^2}{\lambda+n} + b =$$

$$= \frac{1}{2}\ddot{x} - \frac{1}{n}\dot{x}^2 + \frac{1}{2}\frac{1}{n}\dot{x}^2 + \frac{1}{2}\frac{\lambda n\bar{x}^2}{\lambda+n} + b =$$

$$= \frac{1}{2}\ddot{x} - \frac{1}{n}\dot{x}^2 + \frac{1}{2}\frac{1}{n}\dot{x}^2 + \frac{1}{2}\frac{\lambda\dot{x}^2}{n(\lambda+n)} + b = b + \frac{\ddot{x} - (\lambda+n)^{-1}\dot{x}^2}{2}.$$

Now, let's substitute a_n, and b_n from the problem statement:

$$p(x) = (2\pi)^{-n/2}\frac{\sqrt{\lambda}}{(\lambda+n)^{1/2}}\frac{b^a}{\Gamma(a)}\int_0^\infty \sigma^{-2(a_n-1)} e^{-b_n\sigma^{-2}} d\sigma^{-2} = \frac{b^a\Gamma(a_n)\lambda^{\frac{1}{2}}}{(2\pi)^{\frac{n}{2}} b_n^{a_n}\Gamma(a)(\lambda+n)^{\frac{1}{2}}}.$$

Part ii

The marginal likelihood from the previous part for $n = 1$ and one observation t is:

$$p(t) = \frac{b^a \Gamma(a + \frac{1}{2}) \lambda^{\frac{1}{2}}}{(2\pi)^{\frac{1}{2}} (b + \frac{1}{2} \frac{\lambda}{\lambda+1} t^2)^{a+\frac{1}{2}} \Gamma(a) (\lambda + 1)^{\frac{1}{2}}}.$$

$St(t|2a, 0, b(1 + \lambda^{-1})/a)$ has a probability density function:

$$St(t|2a, 0, b(1 + \lambda^{-1})/a) = \frac{\Gamma(\frac{2a+1}{2})}{\sqrt{\pi}\sqrt{2a}\sqrt{\frac{b(1+\lambda^{-1})}{a}}\Gamma(a)} \left(1 + \frac{t^2}{2a\frac{b(1+\lambda^{-1})}{a}}\right)^{-a-\frac{1}{2}}$$

$$= \frac{\Gamma(a + \frac{1}{2})}{\sqrt{b}\left(\frac{\lambda+1}{\lambda}\right)^{\frac{1}{2}} (2\pi)^{\frac{1}{2}} \Gamma(a)} \left(1 + \frac{t^2}{2b(1 + \lambda^{-1})}\right)^{-a-\frac{1}{2}} =$$

$$= \frac{\lambda^{\frac{1}{2}} \Gamma(a + \frac{1}{2})}{\sqrt{b} (\lambda + 1)^{\frac{1}{2}} (2\pi)^{\frac{1}{2}} \Gamma(a)} \left(1 + \frac{t^2}{2b(1 + \lambda^{-1})}\right)^{-a-\frac{1}{2}} =$$

$$= \frac{\lambda^{\frac{1}{2}} \Gamma(a + \frac{1}{2})}{b^{\frac{1}{2}} (\lambda + 1)^{\frac{1}{2}} (2\pi)^{\frac{1}{2}} \Gamma(a)} \left(\frac{2b(1 + \lambda^{-1}) + t^2}{2(1 + \lambda^{-1})}\right)^{-a-\frac{1}{2}} \frac{1}{b^{-a-\frac{1}{2}}} =$$

$$= \frac{b^a \lambda^{\frac{1}{2}} \Gamma(a + \frac{1}{2})}{(\lambda + 1)^{\frac{1}{2}} (2\pi)^{\frac{1}{2}} \Gamma(a)} \left(b + \frac{1}{2} \frac{\lambda t^2}{\lambda + 1}\right)^{-a-\frac{1}{2}} = p(t).$$

Part b

```python
from typing import List

import numpy as np
import scipy.special as sp


def log_marginal_likelihood(x: List[float], a: float, b: float, lambda_: float) -> float:
  n = len(x)  # number of observations
  a_n = a + n / 2
  x_sum = np.sum(x)
  x_sq_sum = np.sum(x ** 2)
  b_n = b + 0.5 * x_sq_sum - 0.5 * (x_sum ** 2) / (lambda_ + n)

  term_1 = a * np.log(b) + sp.loggamma(a_n) + 0.5 * np.log(lambda_)
  term_2 = 0.5 * n * np.log(2 * np.pi) + a_n * np.log(b_n)
  term_2 += sp.loggamma(a) + 0.5 * np.log(lambda_ + n)

  return term_1 - term_2
```

Part c

```
import pandas as pd


df = pd.read_csv("cc_deaths_unc.csv")
df.head()
```

```
##    id  study       rx  sex  age  ...  differ  extent  surg  node4  time
## 0   1      1  Lev+5FU    1   43  ...     2.0       3     0      1  1521
## 1   3      1      Obs    0   71  ...     2.0       2     0      1   963
## 2   4      1  Lev+5FU    0   66  ...     2.0       3     1      1   293
## 3   5      1      Obs    1   69  ...     2.0       3     1      1   659
## 4   6      1  Lev+5FU    0   57  ...     2.0       3     0      1  1767
##
## [5 rows x 14 columns]
```

```
y = df["time"].values
y_prime = np.log(y)
model_1 = log_marginal_likelihood(x=y_prime, a=0.1, b=0.1, lambda_=0.01)
model_1
```

```
## -544.245932746674
```

Part d

Part i

```
from scipy.stats import t


def student_t_distribution(x, a, b, lambda_):
    df = 2 * a  # degrees of freedom
    loc = 0  # location parameter
    scale = np.sqrt(b * (1 + 1 / lambda_) / a)  # scale parameter

    return t.pdf(x, df, loc, scale)


def dp_log_marginal_likelihood(
    x: List[float], alpha: float, a: float, b: float, lambda_: float
) -> None:
    n = len(x)  # number of observations
    term_1 = sp.loggamma(alpha) - sp.loggamma(alpha + n)
    term_2 = 0
    for i in range(n):
        curr_list = x[:i]
        curr_element = x[i]

        distr_term = 0
```

```
        number_of_elements = 0
        if curr_element in curr_list:
            number_of_elements = np.count_nonzero(curr_list == curr_element)
        else:
            distr_term = alpha * student_t_distribution(curr_element, a, b, lambda_)

        term_2 += np.log(distr_term + number_of_elements)

    return term_1 + term_2
```

Part ii

```
model_2 = dp_log_marginal_likelihood(x=y_prime, alpha=1, a=0.1, b=0.1, lambda_=0.01)
model_2
```

```
## -4199.335865193275
```

Part iii

The Bayes Factor in favour of model_1 over model_2 is

$$B_{12}(x) = \frac{p_1(x)}{p_2(x)},$$

or the log Bayes Factor is

$$\log B_{12}(x) = \log p_1(x) - \log p_2(x).$$

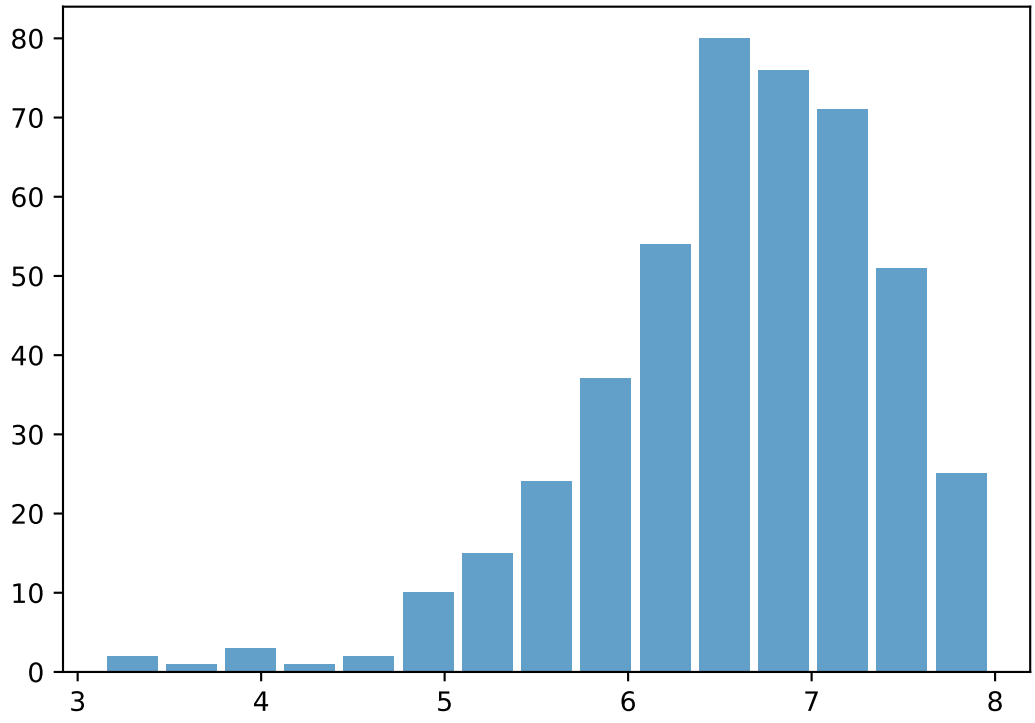
In Python:

```
model_1 - model_2
```

```
## 3655.089932446601
```

Evidence in favor of the normal distribution model is very strong.

```
import matplotlib.pyplot as plt

plt.hist(y_prime, bins=15, alpha=0.7, rwidth=0.85);
plt.show()
```



A possible reason might be that we are dealing with just 452 observations which do not have complex underlying relationships. For larger samples sizes the possibility of having such more complex relationships grows.

Part iv

For a single observed event time, for a single sample from the normal distribution model the marginal likelihood $p(x)$ corresponds to the $St(t|2a, 0, b(1+\lambda^{-1})/a)$ distribution, which is the base probability measure for the DP model. This implies having a Bayes factor ~0, meaning no evidence towards any of the two models.

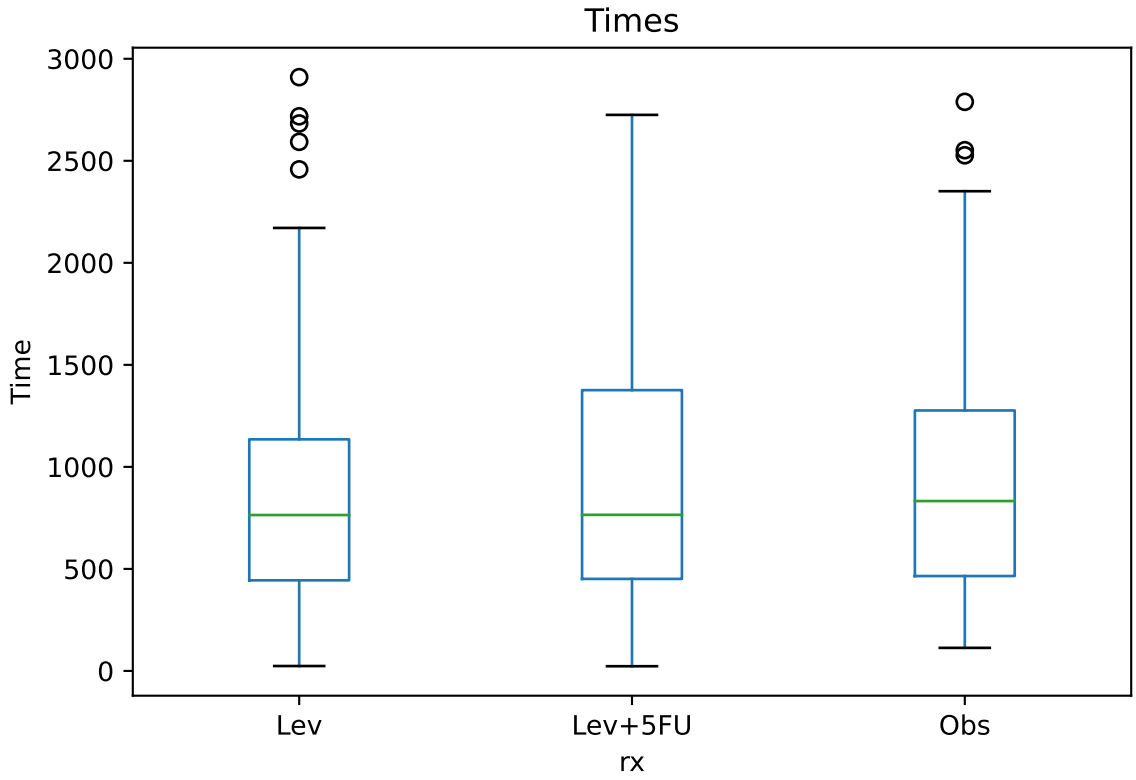
Question 2

Part a

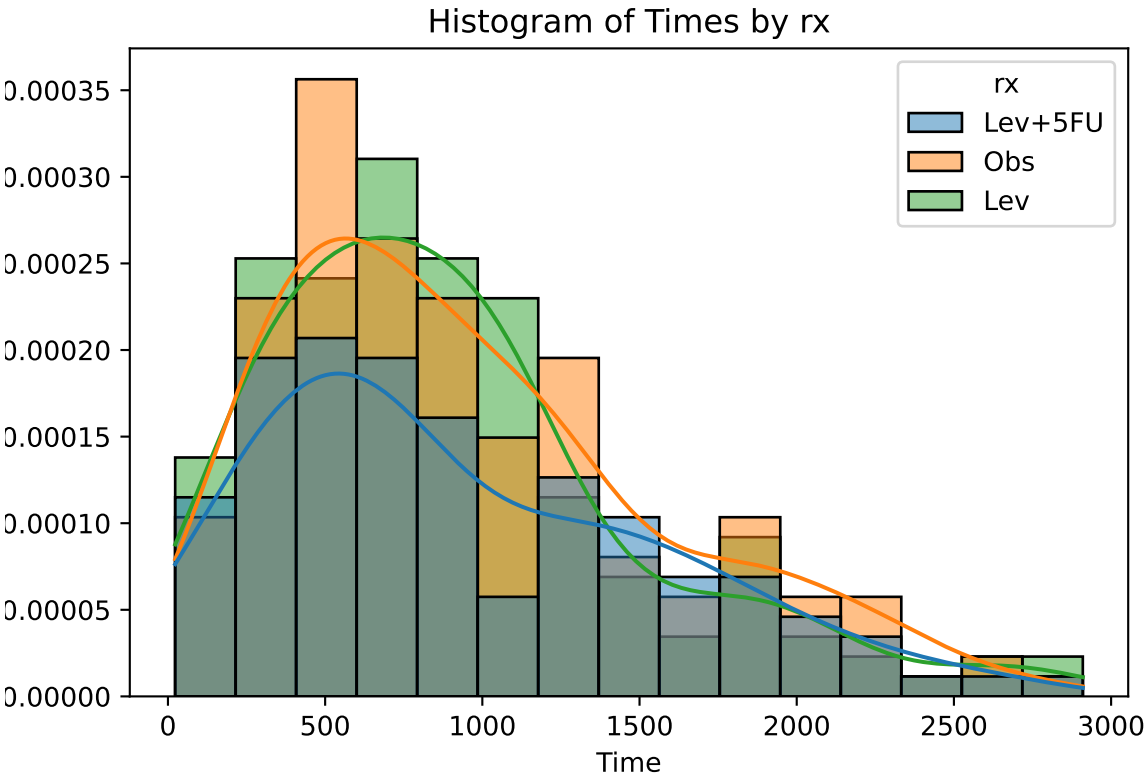
Part i

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt


# box plot:
df.boxplot(column='time', by='rx', grid=False)
```

```
plt.title('Times')
plt.suptitle('')
plt.xlabel('rx')
plt.ylabel('Time')
plt.show()
```




```
# histogram:
sns.histplot(data=df, x='time', hue='rx', kde=True, stat='density')
plt.xlabel('Time')
plt.ylabel('Density')
plt.title('Histogram of Times by rx')
plt.show()
```



There is no significant difference between the three populations. In the `Lev` case we can notice some outliers, in the `Obs` case there are less outliers, and in the `Lev+5FU` case there are no outliers. In the density plot it can be seen that the `Lev+5FU` density is slightly lower up to 1500.

Part ii

```
y_obs = df[df["rx"] == "Obs"]["time"].values
y_lev = df[df["rx"] == "Lev"]["time"].values
y_lev_5fu = df[df["rx"] == "Lev+5FU"]["time"].values

y_obs_prime = np.log(y_obs)
y_lev_prime = np.log(y_lev)
y_lev_5fu_prime = np.log(y_lev_5fu)

model_1_obs = log_marginal_likelihood(x=y_obs_prime, a=0.1, b=0.1, lambda_=0.01)
model_1_lev = log_marginal_likelihood(x=y_lev_prime, a=0.1, b=0.1, lambda_=0.01)
model_1_lev_5fu = log_marginal_likelihood(x=y_lev_5fu_prime, a=0.1, b=0.1, lambda_=0.01)

log_marginal_likelihood_m1 = model_1_obs + model_1_lev + model_1_lev_5fu
log_marginal_likelihood_m1
```

```
## -555.7722624910245
```

```
log_bayes_factor_2 = model_1 - log_marginal_likelihood_m1
np.exp(log_bayes_factor_2)
```

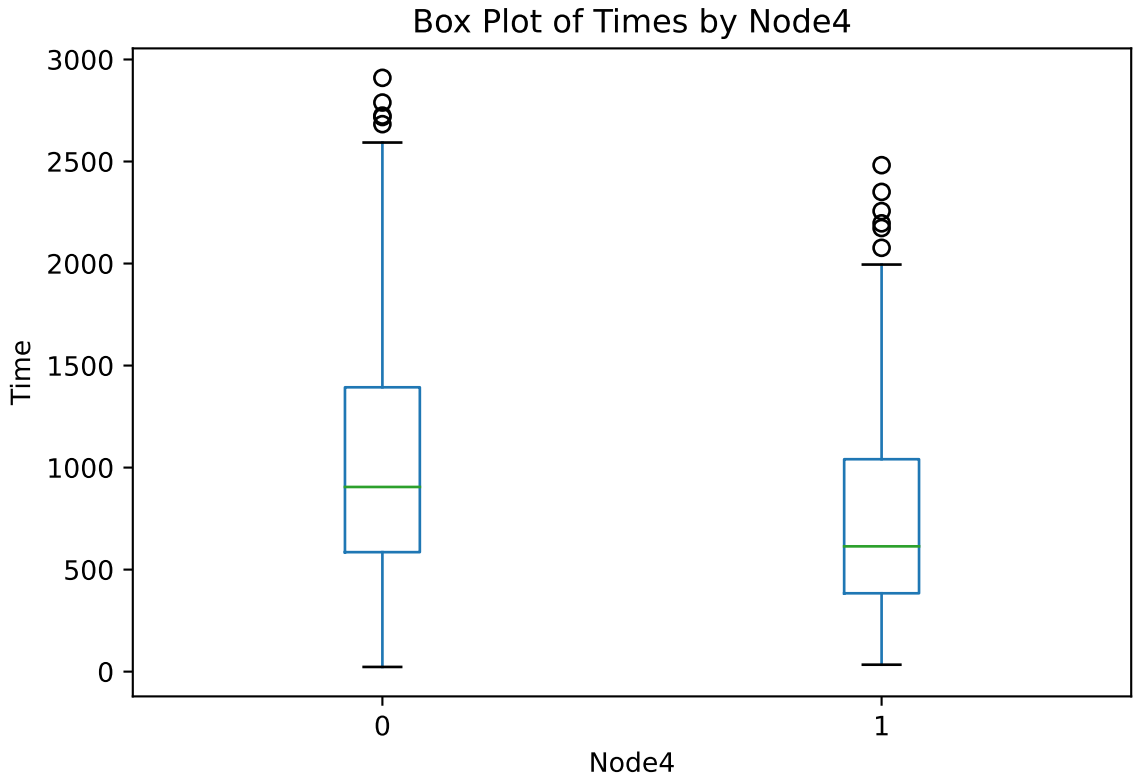
```
## 101349.45194835271
```

Again, there is strong evidence in favour of the normal distribution model. The Bayes factor is smaller than the the previous one comparing the two models on all of the data. This means some relationships were caught by the M1 model fit separately on each of the 3 populations, suggesting there are some distribution distinctions between them. A slight distinction was seen for the combined treatment case.

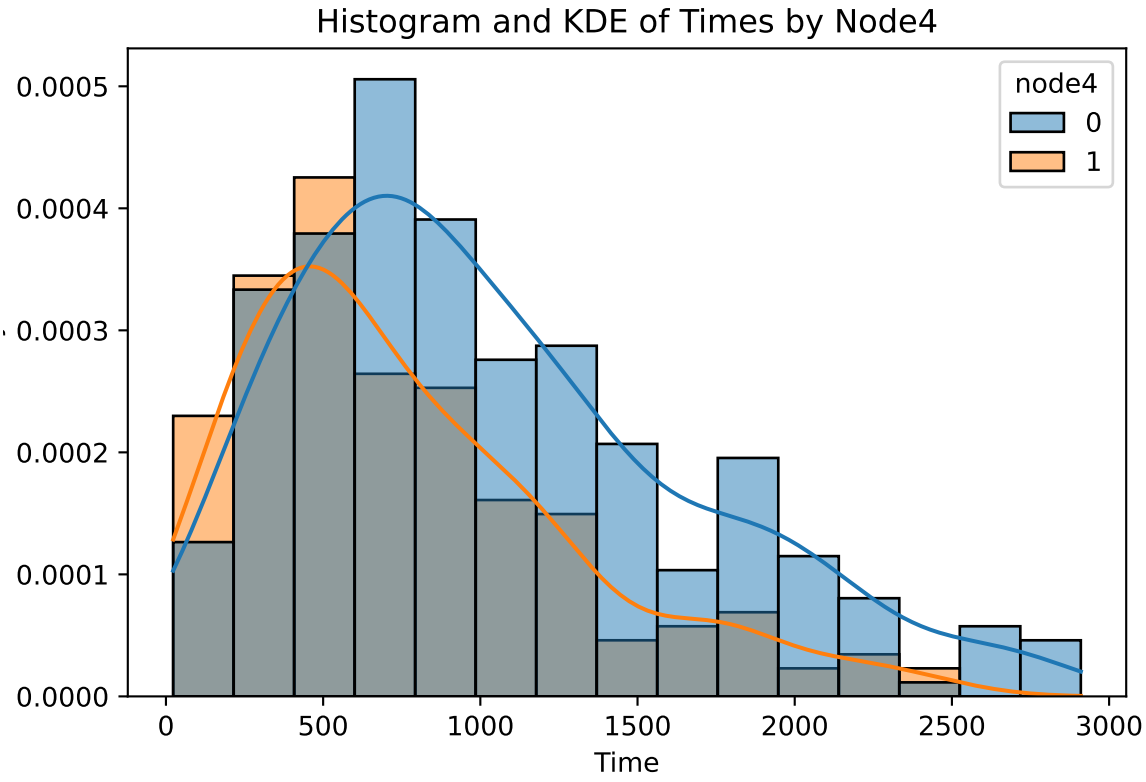
Part b

Part i

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

# color mapping for the different 'node4' values
color_dict = {node: color for node, color in zip(df['node4'].unique(), sns.color_palette("husl", len(df

# box plot
df.boxplot(column='time', by='node4', grid=False)
plt.title('Box Plot of Times by Node4')
plt.suptitle('')
plt.xlabel('Node4')
plt.ylabel('Time')
plt.show()
```




```
# histogram with KDE
sns.histplot(data=df, x='time', hue='node4', kde=True, stat="density")
plt.xlabel('Time')
plt.ylabel('Density')
plt.title('Histogram and KDE of Times by Node4')
plt.show()
```



We can see there are differences in the population means, as well as the maximum possible values.

Part ii

```
y_0 = df[df["node4"] == 0]["time"].values
y_1 = df[df["node4"] == 1]["time"].values

y_0_prime = np.log(y_0)
y_1_prime = np.log(y_1)

model_1_y0 = log_marginal_likelihood(x=y_0_prime, a=0.1, b=0.1, lambda_=0.01)
model_1_y1 = log_marginal_likelihood(x=y_1_prime, a=0.1, b=0.1, lambda_=0.01)

log_marginal_likelihood_m1 = model_1_y0 + model_1_y1
log_marginal_likelihood_m1
```

```
## -542.7653512920415
```

```
log_bayes_factor_3 = model_1 - log_marginal_likelihood_m1
np.exp(log_bayes_factor_3)
```

```
## 0.22750536586885525
```

The Bayes factor is less than 1, suggesting that there is evidence in favour of M1 model over M0. This agress with the visual interpretations.The result suggest there is not significant difference between fitting the model on all of data or separately.

Question 3

Part a

Part i

```
stan_model_code = """
data {
  int<lower=0> n;          // number of observations
  int<lower=1> p;          // number of covariates
  matrix[n, p] x;          // covariate matrix
  vector[n] y;             // response vector
  real<lower=0> xi;        // standard deviation parameter
}

parameters {
  vector[p] beta;          // regression coefficients
}

model {
  // prior distribution for beta
  beta ~ normal(0, xi);

  // likelihood
  for (i in 1:n) {
    y[i] ~ exponential(exp(x[i] * beta));
  }
}
"""
```

Part ii

Simulating data for the model:

```
from scipy.stats import expon

np.random.seed(55)

# Set parameters
n = 1000
p = 1
xi = 10
beta_true = np.array([2.0])

# Generate predictor values
x = np.random.normal(size=(n, p))
```

```
# Calculate rate parameters
lambda_i = np.exp(x @ beta_true)
# Generate response values
y = expon.rvs(scale=1/lambda_i)
```

Fit PyStan model:

```
import stan


stan_data = {'n': n, 'p': p, 'x': x, 'y': y, 'xi': xi}
stan_model = stan.build(stan_model_code, data=stan_data, random_seed=1)
```

```
# Fit Stan model
chains, samples, burn = 1, 10_000, 1000
stan_fit = stan_model.sample(num_chains=chains, num_samples=samples, num_warmup=burn, save_warmup=False
```

```
res = stan_fit.to_frame()
res
```

```
## parameters        lp__  accept_stat__  ...    energy__    beta.1
## draws                                  ...
## 0          -980.785259       0.997028  ...  980.825680  2.002018
## 1          -980.785259       0.925436  ...  981.227501  2.002018
## 2          -980.759759       0.997628  ...  980.803931  2.010854
## 3          -980.765013       0.998104  ...  980.774282  2.005694
## 4          -981.367238       0.885153  ...  981.569444  1.974649
## ...                ...            ...  ...         ...       ...
## 9995       -980.893272       1.000000  ...  980.955718  2.025559
## 9996       -980.796368       0.995051  ...  980.945801  2.017909
## 9997       -980.783816       0.925849  ...  981.407709  2.016335
## 9998       -980.758921       0.998365  ...  980.796846  2.008360
## 9999       -980.844683       0.989617  ...  980.852415  1.996250
##
## [10000 rows x 8 columns]
```

Part iii

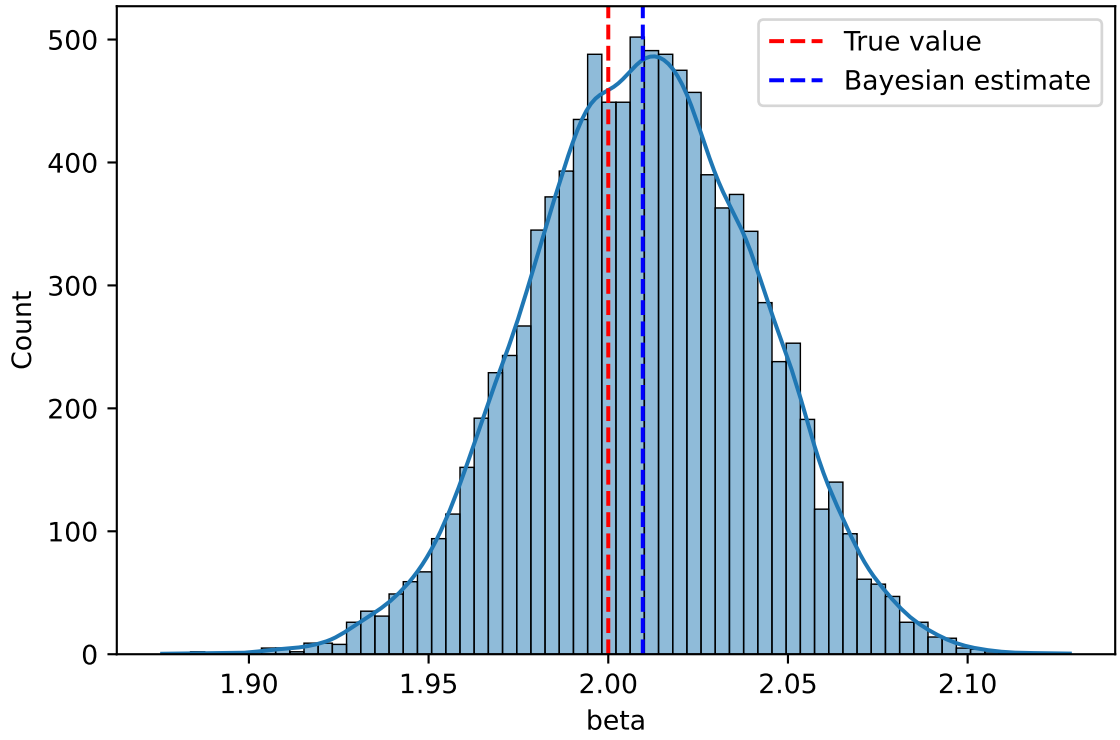
```
import matplotlib.pyplot as plt
import seaborn as sns

# Get the samples from the posterior
res = res.rename(columns={"beta.1": "beta"})

# Calculate the Bayesian estimate of beta (posterior mean)
bayesian_estimate = np.mean(res['beta'])

# Compare the estimate with the true value
print(f"True value: {beta_true[0]}")
```

```
## True value: 2.0
```

```
print(f"Bayesian estimate: {bayesian_estimate}")
```

```
## Bayesian estimate: 2.0096115041041505
```

```
# posterior distribution of beta
sns.histplot(res['beta'], kde=True)
plt.axvline(beta_true[0], color='r', linestyle='--', label='True value')
plt.axvline(bayesian_estimate, color='b', linestyle='--', label='Bayesian estimate')
plt.legend()
plt.show()
```



Part b

Part i

```
df['rx_binary'] = (df['rx'] != 'Obs').astype(int)

n = len(df)
p = 3 # intercept, node4, treatment_binary
x = pd.concat([pd.Series(np.ones(n), name='intercept'), df[['node4', 'rx_binary']]], axis=1).values
print(x)
```

```
## [[1. 1. 1.]
##  [1. 1. 0.]
##  [1. 1. 1.]
##  ...
##  [1. 0. 1.]
##  [1. 0. 1.]
##  [1. 0. 1.]]
```

```
y = list(df['time'])
xi = 10

stan_data = {'n': n, 'p': p, 'x': x, 'y': y, 'xi': xi}
```

```
# Build and compile the model
stan_model = stan.build(stan_model_code, data=stan_data, random_seed=1)
```

```
# Fit Stan model
chains, samples, burn = 4, 10_000, 1000
stan_fit = stan_model.sample(num_chains=chains, num_samples=samples, num_warmup=burn, save_warmup=False)
```

```
import arviz as az
print(az.summary(stan_fit))
```

```
##           mean     sd  hdi_3%  hdi_97%  ...  mcse_sd  ess_bulk  ess_tail  r_hat
## beta[0] -6.962  0.086  -7.124   -6.800  ...    0.000   16501.0   20017.0    1.0
## beta[1]  0.299  0.097   0.124    0.488  ...    0.000   21467.0   21618.0    1.0
## beta[2]  0.029  0.099  -0.155    0.215  ...    0.001   17986.0   20677.0    1.0
##
## [3 rows x 9 columns]
```

An R-hat value close to 1 indicates good convergence, and we can see just that in the results.

Part ii

```
res = stan_fit["beta"]
print(np.mean(res > 0, axis=-1))
```

```
## [0.      0.99885 0.61355]
```

- $\beta_1 = 0.999$ indicates that Node4 has a high positive contribution to the model outcome
- $\beta_2 = 0.61$ is lower than β_1 suggesting a slightly positive contribution to the outcome (uncensored death times)

This aligns with the findings from Q2 to some extent.