Statistical Modelling Assignment 1

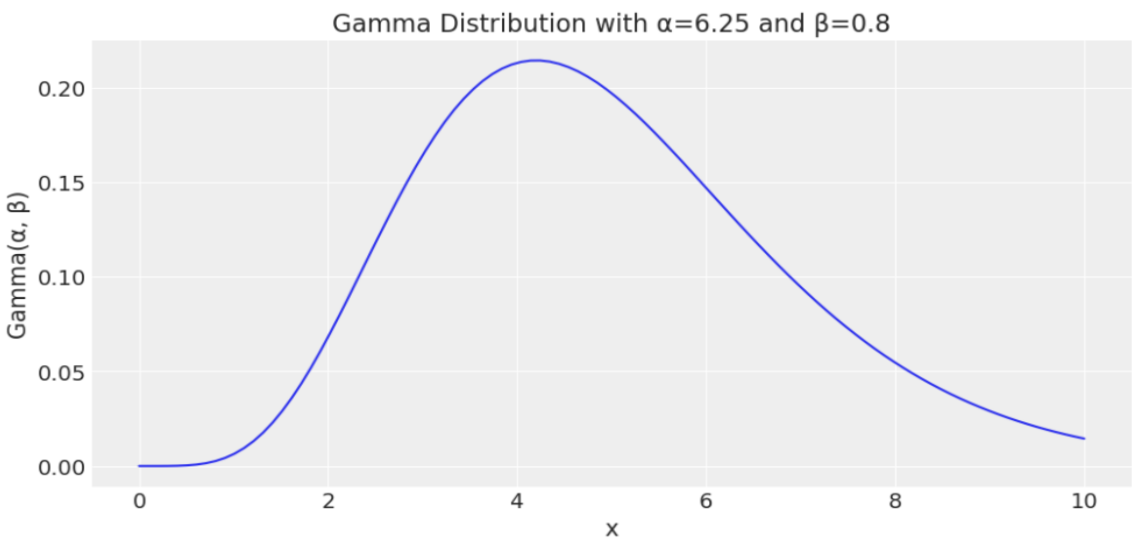
Mahad Ahmed 20i – 0426 Section F

**Question 1**

In this Question, I decided to use the Gamma and Normal prior distributions. For the Gamma distribution, the mean was randomly generated between 3 and 8 , while the standard deviation was set to 2. Using these values, an appropriate formula for Alpha and Beta parameters was found through trial and error which helped satisfy the given conditions in the question. The probability density function of the distribution was calculated via gamma.pdf (x, a, scale=b). The mean and standard deviation were confirmed by numerical methods of sampling 10 000 random variables from the distribution and computing the mean and standard deviation from them. Below is the code and plot for the gamma distribution.

```
#Prior Distribution 1

μ=random.randint(3,8)
σ=2

α=(μ/σ)**2
β=μ/α

# Probability Density Function for Gamma Distribution
x = np.linspace(0,10,100)
y=gamma.pdf(x,α,scale=β)

# Verifying the Distribution
samples=10000
data=gamma(α,scale=β).rvs(samples)
print("Verifying Mean from 10 000 Samples: ",data.mean())
print("Verifying Std from 10 000 Samples: ",data.std())

# Plotting The Distribution
plt.figure(figsize=(10,5))
plt.plot(x,y)
plt.xlabel('x')
plt.ylabel('Gamma(α, β)')
plt.title(f'Gamma Distribution with α={α} and β={β}')
plt.show()
```

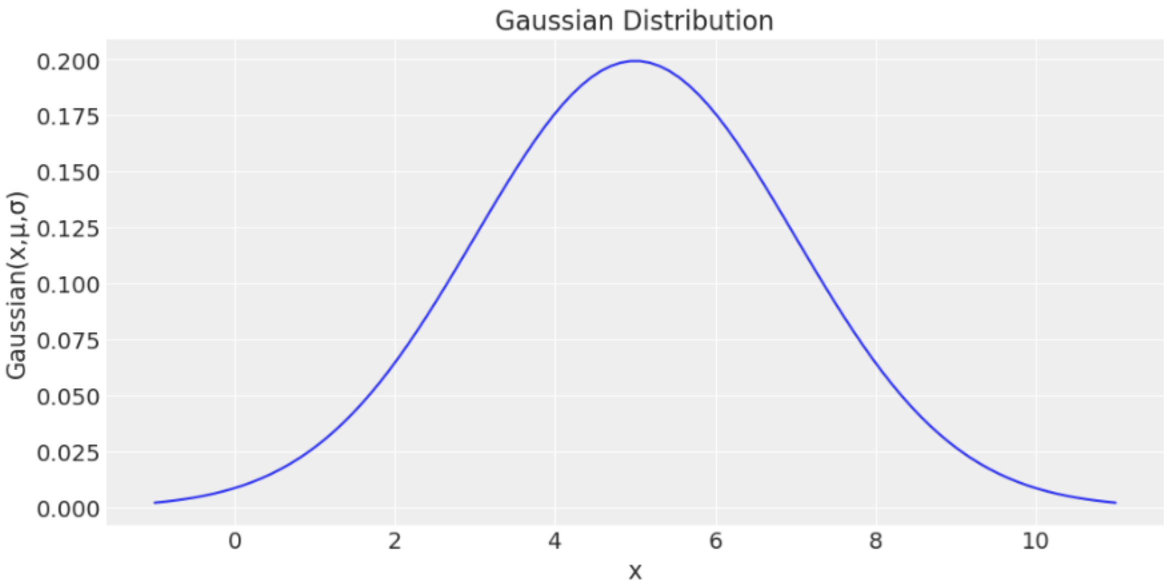
```
Verifying Mean from 10 000 Samples:  5.010786440319066
Verifying Std from 10 000 Samples:  2.038943490823259
```



For the second distribution, I used a normal distribution. The mean and standard deviation were set like the previous distribution. The probability density function of the distribution was found using norm.pdf(x,u,s) . Similar to the gamma distribution the samples were verified and the distribution was plotted. Below is the attached code and plot.

```
#Prior Distribution 2

μ=random.randint(3,8)
σ=2

# Probability Density Function for Gaussian Distribution
x = np.linspace(μ-3*σ,μ+3*σ,100)
y=norm.pdf(x,μ,σ)

# Verifying the Distribution
samples=10000
data=norm(μ,σ).rvs(samples)
print("Verifying Mean from 10 000 Samples: ",data.mean())
print("Verifying Std from 10 000 Samples: ",data.std())

#cPlotting the Distribution
plt.figure(figsize=(10,5))
plt.plot(x,y)
plt.xlabel('x')
plt.ylabel('Gaussian(x,μ,σ)')
plt.title(f'Gaussian Distribution')
plt.show()
```

```
Verifying Mean from 10 000 Samples:  4.979702593779999
Verifying Std from 10 000 Samples:  2.0055296864491052
```




**Question 2**

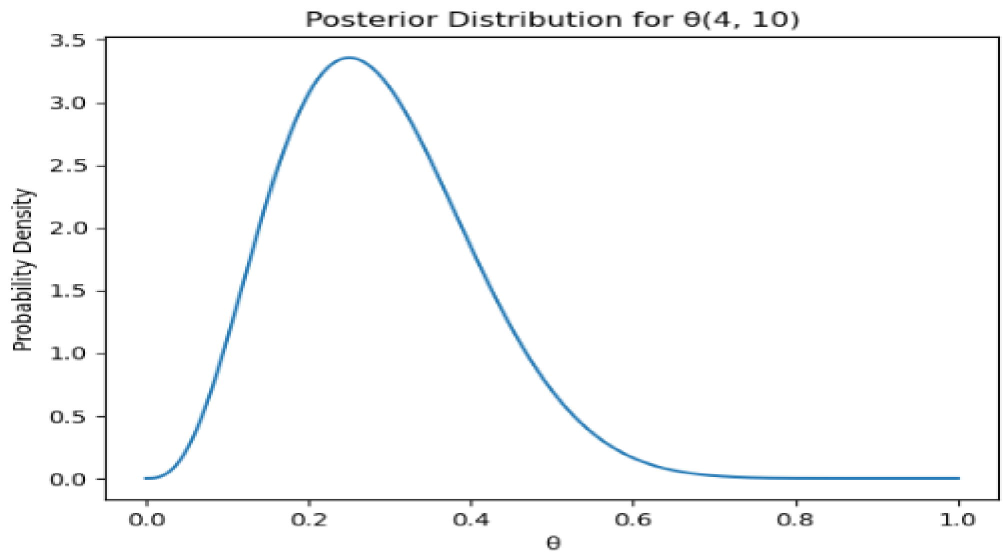
First, I set the total trials, successes, and alpha/beta parameters. Then the posterior was calculated using beta distribution in which we passed alpha + number of successes and beta + number of failures as parameter.

For Part A, the equal tailed interval was calculated using posterior.ppf function in which 0.03 and 0.97 bounded the interval and were sent as arguments to ppf function. The HDI was calculated by getting the probability density function. To see the 94%, we iteratively added the cumulative density until it reached 94% while adding the x values in a list. The minimum and maximum values in this list represents our interval for the HDI.

For part B, 15 000 samples were generated using this beta distribution. Next, the Predictive posterior was found by using binomial. By finding the mean of the predictive posterior on the condition that the predictive posterior is greater than 1 (More than one germinated), the probability of at least 1 seed germinating was found.

```python
# Choosing the Prior and getting Posterior
n=12
y=3
a_prior=1
b_prior=1
p_theta_given_y=beta(a_prior+y,b_prior+n-y)

#Part A -----------------------

#Equal Tailed Interval
eti=(p_theta_given_y.ppf(0.03),p_theta_given_y.ppf(0.97))
print(f"Equal Tailed Interval: {eti[0]:.3f},{eti[1]:.3f} ")

#Highest Density 94% Interval
x=np.linspace(0,1,1000)
pdf=beta.pdf(x,a_prior+y,b_prior+n-y)
pdf=pdf/pdf.sum()
indexes=pdf.argsort()[::-1]
clt=0 # cumulative
vals_HDI=[]
for i in indexes:
    clt+=pdf[i]
    if clt>0.94:
        break
    else:
        vals_HDI.append(x[i])
print(f"Highest Density 94% Intervals: {min(vals_HDI):.3f},{max(vals_HDI):.3f}")

pdf=beta.pdf(x,a_prior+y,b_prior+n-y)
plt.plot(x,pdf)
plt.xlabel("θ")
plt.ylabel("Probability Density")
plt.title(f"Posterior Distribution for θ({a_prior+y}, {b_prior+n-y})")
```

```
Equal Tailed Interval: 0.096,0.527
Highest Density 94% Intervals: 0.080,0.503

Text(0.5, 1.0, 'Posterior Distribution for θ(4, 10)')
```




```python
# Part B

samples=15000
probabilities=beta(a_prior+y,b_prior+n-y).rvs(samples)

posterior_predictive = stats.binom(p=probabilities,n=12).rvs()

prob_of_1=np.mean(posterior_predictive>=1)
print(f"Probability Of Atleast One is: {prob_of_1:.3f}")
```

```
Probability Of Atleast One is: 0.941
```