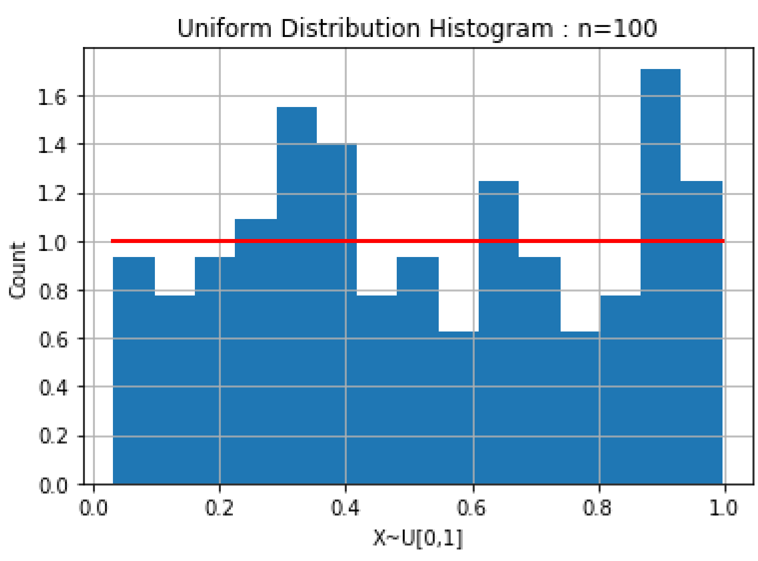
Problem 1

Sample n=100, 1000, 10000 points from (a) Uniform Distribution in 0 to 1. (b) Gaussian Distribution with mean 0 and variance 1. (c) Exponential Distribution with rate parameter = 1. Verify if the points are generated according to the respective distribution by plotting a histogram of the fraction of points in each case. Label graph properly. Use inbuilt library functions for sampling from each of the above mentioned distributions.

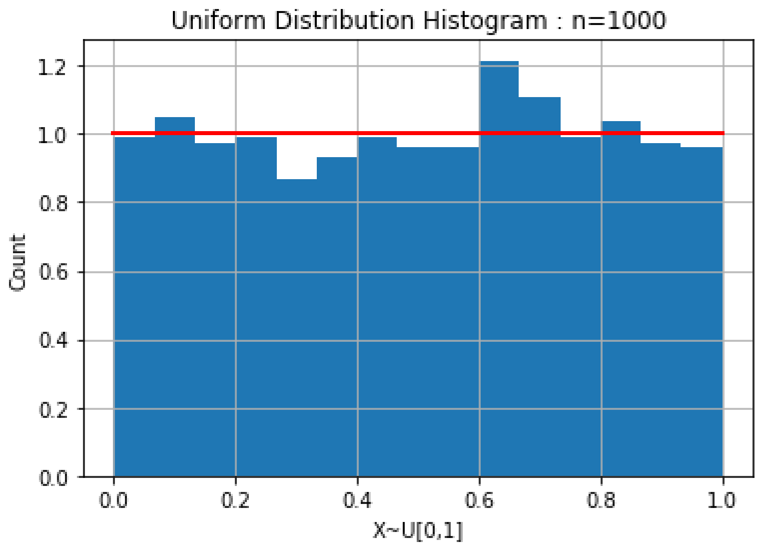
Uniform distribution in 0 to 1

```
import numpy as np
import matplotlib.pyplot as plt
s = np.random.uniform(0,1,100)
count, bins, ignored = plt.hist(s, 15, density=True)
plt.plot(bins, np.ones_like(bins), linewidth=2, color='r')
plt.xlabel('X~U[0,1]')
plt.ylabel('Count')
plt.title('Uniform Distribution Histogram : n=100')
plt.grid(True)

plt.show()
```

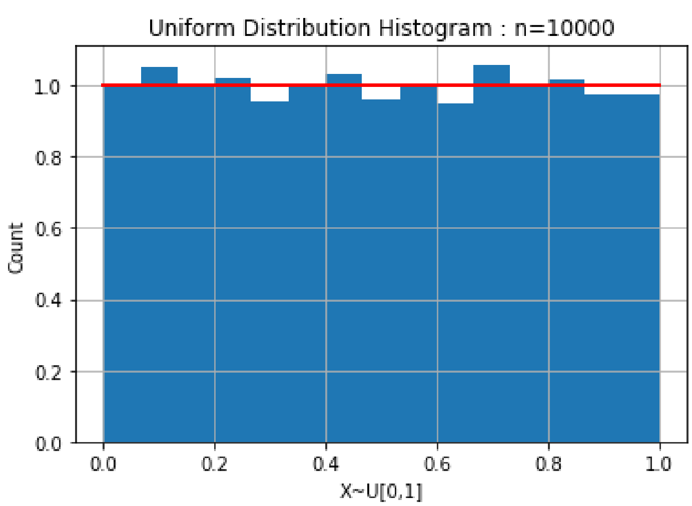



```
import numpy as np
import matplotlib.pyplot as plt
s = np.random.uniform(0,1,1000)
count, bins, ignored = plt.hist(s, 15, density=True)
plt.plot(bins, np.ones_like(bins), linewidth=2, color='r')
```

```
plt.xlabel('X~U[0,1]')
plt.ylabel('Count')
plt.title('Uniform Distribution Histogram : n=1000')
plt.grid(True)

plt.show()
```

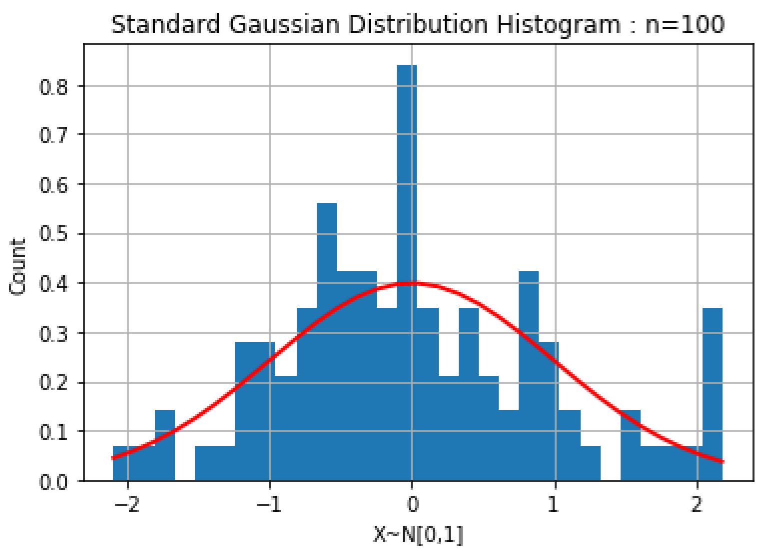



```
import numpy as np
import matplotlib.pyplot as plt
s = np.random.uniform(0,1,10000)
count, bins, ignored = plt.hist(s, 15, density=True)
plt.plot(bins, np.ones_like(bins), linewidth=2, color='r')
plt.xlabel('X~U[0,1]')
plt.ylabel('Count')
plt.title('Uniform Distribution Histogram : n=10000')
plt.grid(True)

plt.show()
```



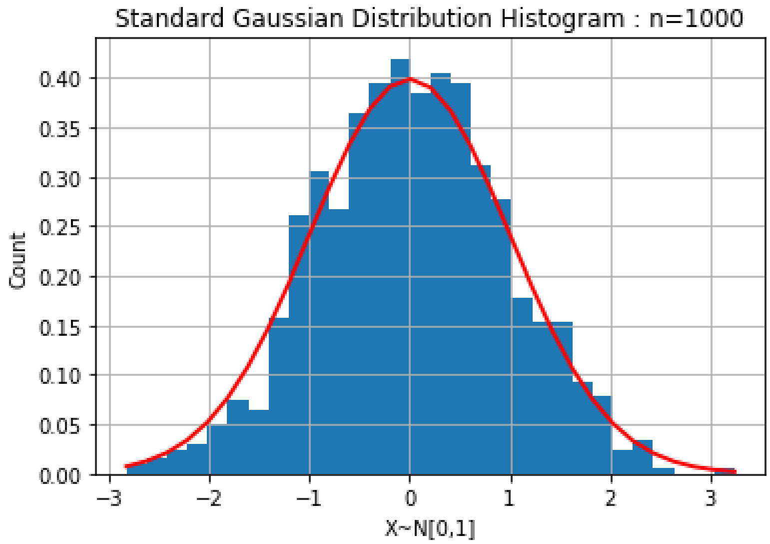
Gaussian Distribution with mean 0 and variance 1

```
import numpy as np
import matplotlib.pyplot as plt
mu,sigma=0,1
s = np.random.normal(mu, sigma, 100)
count, bins, ignored = plt.hist(s, 30, density=True)
plt.plot(bins, 1/(sigma * np.sqrt(2 * np.pi)) *
             np.exp( - (bins - mu)**2 / (2 * sigma**2) ),
        linewidth=2, color='r')
plt.xlabel('X~N[0,1]')
plt.ylabel('Count')
plt.title('Standard Gaussian Distribution Histogram : n=100')
plt.grid(True)

plt.show()
```

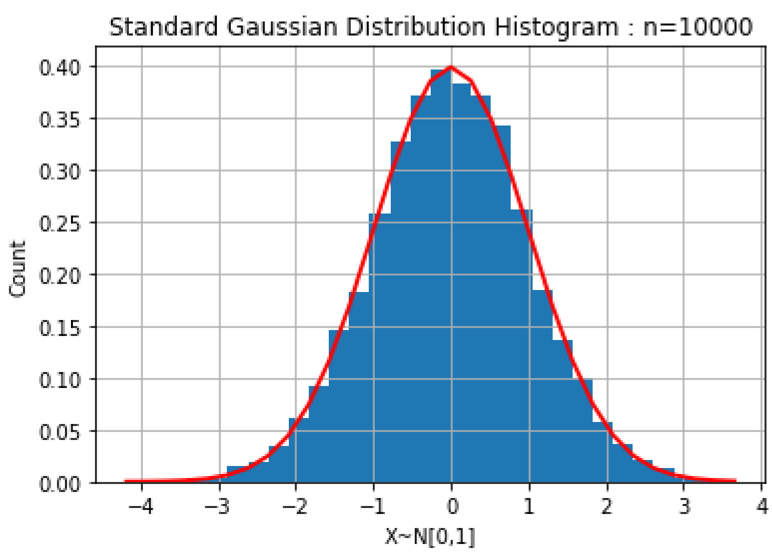



```
import numpy as np
import matplotlib.pyplot as plt
mu,sigma=0,1
s = np.random.normal(mu, sigma, 1000)
count, bins, ignored = plt.hist(s, 30, density=True)
plt.plot(bins, 1/(sigma * np.sqrt(2 * np.pi)) *
             np.exp( - (bins - mu)**2 / (2 * sigma**2) ),
        linewidth=2, color='r')
plt.xlabel('X~N[0,1]')
plt.ylabel('Count')
plt.title('Standard Gaussian Distribution Histogram : n=1000')
plt.grid(True)
```

plt.show()

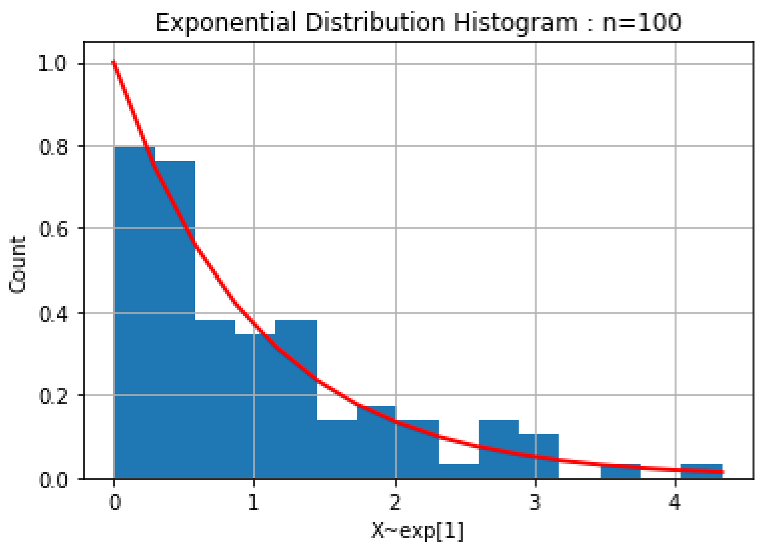



```
import numpy as np
import matplotlib.pyplot as plt
mu,sigma=0,1
s = np.random.normal(mu, sigma, 10000)
count, bins, ignored = plt.hist(s, 30, density=True)
plt.plot(bins, 1/(sigma * np.sqrt(2 * np.pi)) *
             np.exp( - (bins - mu)**2 / (2 * sigma**2) ),
        linewidth=2, color='r')
plt.xlabel('X~N[0,1]')
plt.ylabel('Count')
plt.title('Standard Gaussian Distribution Histogram : n=10000')
plt.grid(True)

plt.show()
```

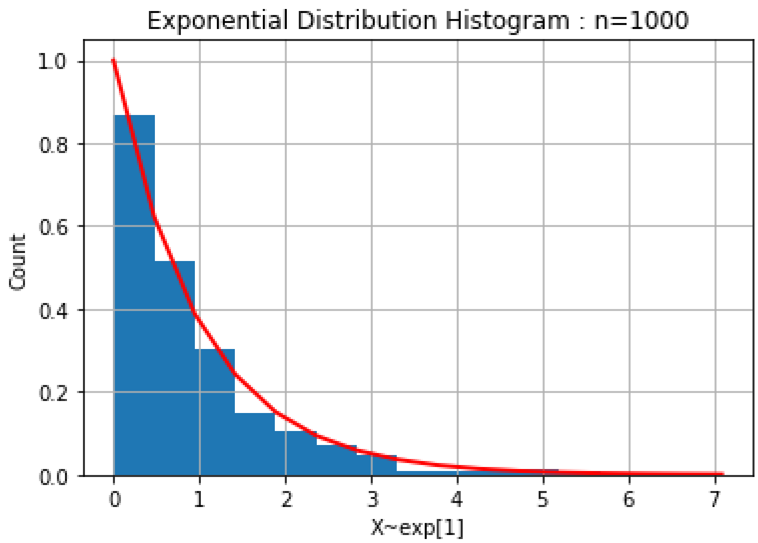


Exponential Distribution with rate parameter 1

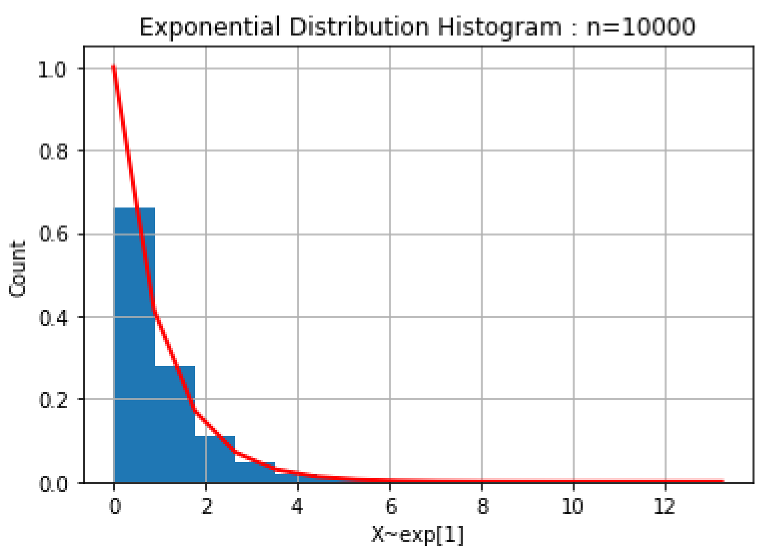
```
import numpy as np
import matplotlib.pyplot as plt
beta=1 #beta=1/lambda
s = np.random.exponential(beta,100)
count, bins, ignored = plt.hist(s, 15, density=True)
plt.plot(bins, 1/(beta)*np.exp(-(bins)/(beta)) , linewidth=2, color='r')
plt.xlabel('X~exp[1]')
plt.ylabel('Count')
plt.title('Exponential Distribution Histogram : n=100')
plt.grid(True)

plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
beta=1 #beta=1/lambda
s = np.random.exponential(beta,1000)
count, bins, ignored = plt.hist(s, 15, density=True)
plt.plot(bins, 1/(beta)*np.exp(-(bins)/(beta)) , linewidth=2, color='r')
plt.xlabel('X~exp[1]')
plt.ylabel('Count')
plt.title('Exponential Distribution Histogram : n=1000')
plt.grid(True)

plt.show()
```



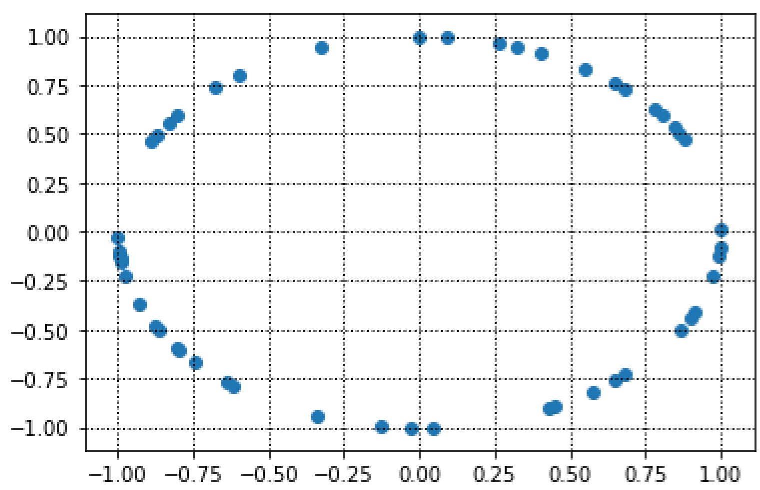

```
import numpy as np
import matplotlib.pyplot as plt
beta=1 #beta=1/lambda
s = np.random.exponential(beta,10000)
count, bins, ignored = plt.hist(s, 15, density=True)
plt.plot(bins, 1/(beta)*np.exp(-(bins)/(beta)) , linewidth=2, color='r')
plt.xlabel('X~exp[1]')
plt.ylabel('Count')
plt.title('Exponential Distribution Histogram : n=10000')
plt.grid(True)

plt.show()
```



Problem 2

Assume a unit circle centered at (0,0). Let n = 2, 3, 5 points be uniformly sampled from the circumference of the circle. Write a python program to estimate the probability of n = 2, 3, 5 points lie within some semi-circle. Verify your answer by solving the question analytically.

```
import numpy as np
import matplotlib.pyplot as plt
def generate_point(n):
    global theta
    theta=np.random.uniform(0,np.pi*2,n)
    #print(theta)
    #plt.hist(theta)
    points=[(np.cos(theta[i]),np.sin(theta[i])) for i in range(n)]
    return points
a=generate_point(50)
x,y=zip(*a)
#print(x)
#print(y)
plt.scatter(x,y)
plt.grid(color='k', linestyle=':', linewidth=1)
#plt.axes().set_aspect('equal', 'datalim')
plt.show()
```



```
def estm_prob(n):
  count=0
  for i in range(50000):
    generate_point(n)
    theta_sorted=np.sort(theta)
    if (max(theta_sorted)-min(theta_sorted))<=np.pi:
      count+=1
    elif (max(theta_sorted)-min(theta_sorted))>np.pi:
      for j in theta_sorted[1:]:
        if (j-theta_sorted[0]<np.pi):
          break
        else:
          pass
        count+=1
  return count/50000

def true_prob(n):
  return n/2**(n-1)

#when 2 points are sampled
a=true_prob(2)
b=estm_prob(2)
print("For n=2, True Probability=",a,"and Estimated Probability=",b)

#when 2 points are sampled
c=true_prob(3)
d=estm_prob(3)
print("For n=2, True Probability=",c,"and Estimated Probability=",d)

#when 2 points are sampled
e=true_prob(5)
f=estm_prob(5)
print("For n=2, True Probability=",e,"and Estimated Probability=",f)
```

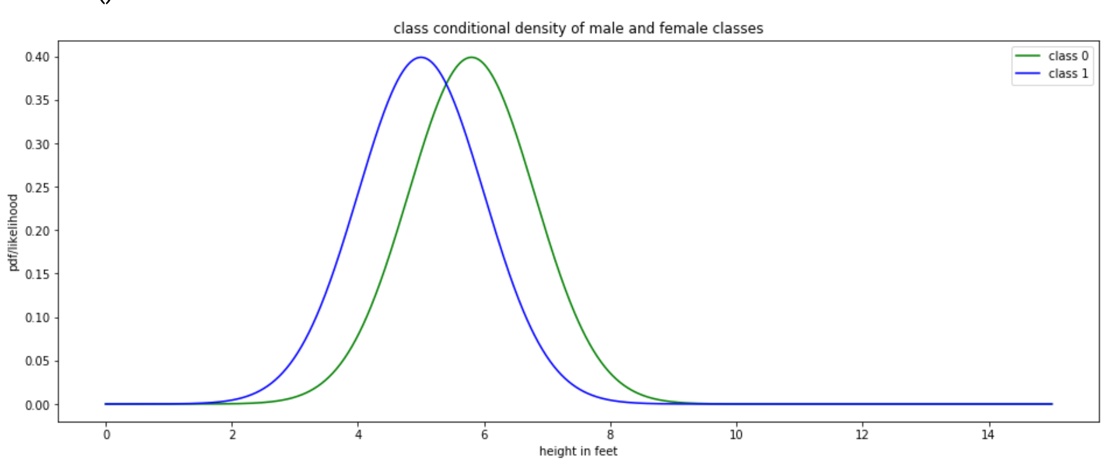
Output:

For n=2, True Probability= 1.0 and Estimated Probability= 1.0
For n=2, True Probability= 0.75 and Estimated Probability= 0.748
For n=2, True Probability= 0.3125 and Estimated Probability= 0.319

Problem 3

Assume two classes male and female. The height of the male class is distributed according to the normal distribution with a mean of 5.8 feet and a standard deviation of 1 feet and the height of the female class is distributed with a mean of 5 feet and a standard deviation of 1 feet. Assume following prior probabilities for two classes (a) For Male 0.5 and for Female 0.5. (b) For Male 0.1 and for Female 0.9. For each of the above cases, specify the priors, plot the class conditional densities and posterior probabilities of both the classes. What is Baye's classifier? Compute the misclassification error. Draw the decision boundary for the Bayes classifier.

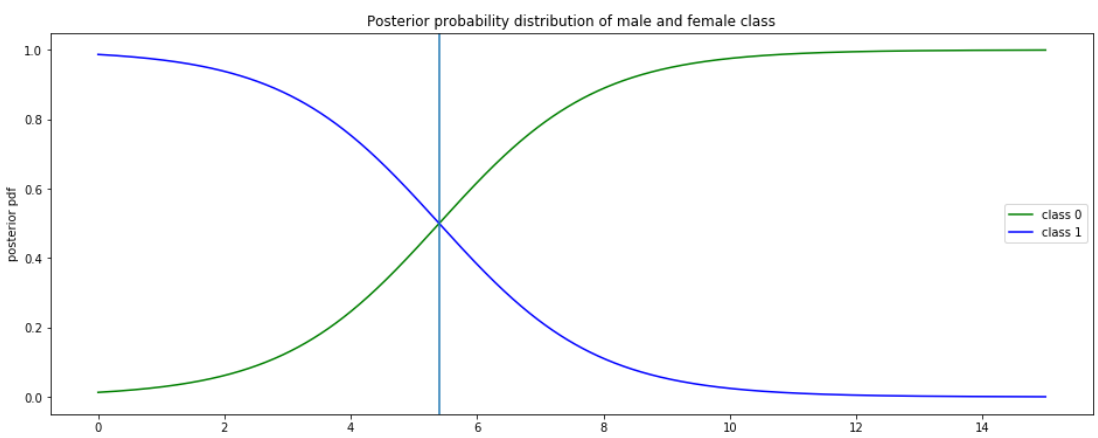
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
#male are class 0 and female are class 1
male_mean=5.8
male_sd=1
female_mean=5
female_sd=1
x=np.linspace(0,15,1000)
f0_X=norm.pdf(x,male_mean,male_sd)
f1_X=norm.pdf(x,female_mean,female_sd)
plt.figure(figsize=(16,6))
plt.plot(x,f0_X,'g')
plt.plot(x,f1_X,'b')
plt.legend(['class 0','class 1'])
plt.title('class conditional density of male and female classes')
plt.xlabel('height in feet')
plt.ylabel('pdf/likelihood')
plt.show()
```



```
#case 1
p0=0.5 #prior probability of male class
p1=0.5 #prior probability of female class
fx_X=[]
q0=[] #posterior probability for class 0
q1=[] #posterior probability for class 1
for i in range(1000):
   fx_X.append(f0_X[i]*p0+f1_X[i]*p1)
   q0.append(f0_X[i]*p0/fx_X[i])
   q1.append(f1_X[i]*p1/fx_X[i])

dec_boundary=(male_mean+female_mean)/2
print('decision boundary for Bayes Classifier=',dec_boundary,'feet')
plt.figure(figsize=(16,6))
plt.plot(x,q0,'g')
plt.plot(x,q1,'b')
plt.axvline(dec_boundary)
plt.legend(['class 0','class 1'])
plt.title('Posterior probability distribution of male and female class')
plt.xlabel('height in feet')
plt.ylabel('posterior pdf')

plt.show()

error=p0*norm.cdf(dec_boundary,loc=male_mean,scale=male_sd)+p1*norm.sf(dec_boundary,loc=female
_mean,scale=female_sd)
print('Misclassification error=',error)
```

Output

decision boundary for Bayes Classifier= 5.4 feet

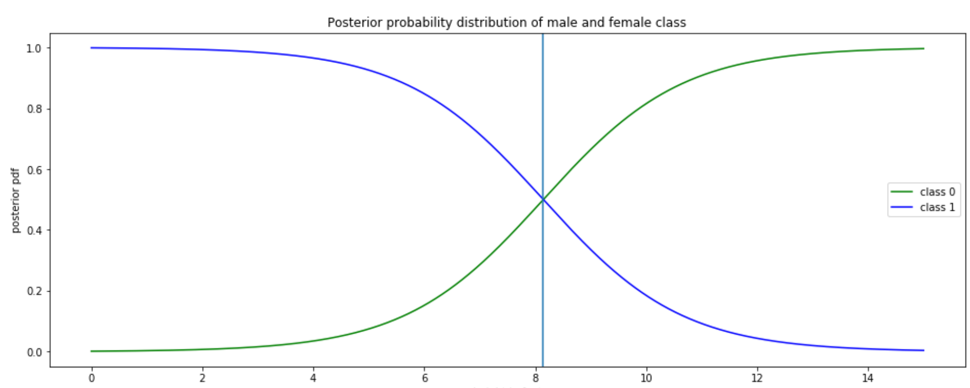


Misclassification error= 0.3445782583896759

```
#case 2
p0=0.1 #prior probability of male class
p1=0.9 #prior probability of female class
fx_X=[]
q0=[] #posterior probability for class 0
q1=[] #posterior probability for class 1
for i in range(1000):
   fx_X.append(f0_X[i]*p0+f1_X[i]*p1)
   q0.append(f0_X[i]*p0/fx_X[i])
   q1.append(f1_X[i]*p1/fx_X[i])

dec_boundary=((male_mean+female_mean)/2)-((male_sd**2)*np.log(p1/p0))/(female_mean-male_mean)
print('decision boundary for Bayes Classifier=',dec_boundary,'feet')
plt.figure(figsize=(16,6))
plt.plot(x,q0,'g')
plt.plot(x,q1,'b')
plt.axvline(dec_boundary)
plt.legend(['class 0','class 1'])
plt.title('Posterior probability distribution of male and female class')
plt.xlabel('height in feet')
plt.ylabel('posterior pdf')

plt.show()

error=p0*norm.cdf(dec_boundary,loc=male_mean,scale=male_sd)+p1*norm.sf(dec_boundary,loc=female
_mean,scale=female_sd)
print('Misclassification error=',error)
```

Output

decision boundary for Bayes Classifier= 8.146530721670276 feet



Misclassification error= 0.0997960343653087

problem-1

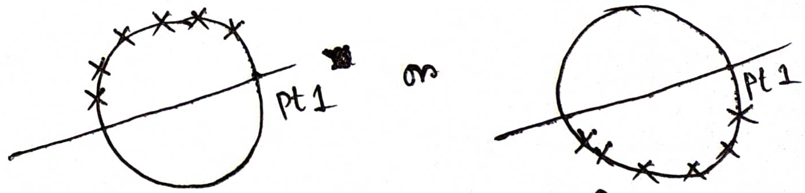
We have a circle at centre $(0,0)$ and radius $= 1$.
We are uniformly sampling $n = 2, 3, 5$ points on the circumferance of the circle.

To calculate
$\mathbb{P}$('n' points lie on the same semi-circle)

for $n = 2$, it is a trivial case. i.e we can always pick two points that lie on the same semi-circle. $\Rightarrow \mathbb{P}(n=2) = 1$.

for $n > 2$,
We will go like this:
1. pick any point on the circumferance of the circle.
2. Draw a diameter through that point.
3. Our desired event is either ~~all~~ rest $(n-1)$ points lie on one side of the diameter.

Now, there are two cases.



So, probability of a point lying one side of diameter is $1/2$.

$\mathbb{P}$(all $(n-1)$ pts lie on the same semi-circle)
$= 1/2 \times 1/2 \times$ —— $(n-1)$ times $= \dfrac{1}{2^{n-1}}$

We can choose pt 1 in $^{n}C_1 = n$ ways.

$\Rightarrow$ ~~total~~ $\mathbb{P}$('n' pts lie on the same semi-circle) $= \dfrac{n}{2^{n-1}}$

$\mathbb{P}(n=2) = \dfrac{1}{1} = 1$, $P(n=3) = \dfrac{3}{4} = 0.75$

$P(n=5) = \dfrac{5}{16} = 0.3125$. (Ans) problem-3

We are given height distribution (that is our feature x) of male and female classes.

So, there are two classes: male — class 0
female — class 1

~~ma~~ class conditional density:

male : class 0 : $f_0(x) \sim N(5.8, 1) = f(x/0)$ $\mu_0 = 5.8$

female : class 1 : $f_1(x) \sim N(5, 1) = f(x/1)$ $\mu_1 = 5$

$\sigma = 1$

Case A :-

prior probabilities :-

$P_0 = P(\text{class}=\text{male}) = 0.5$

$P_1 = P(\text{class}=\text{female}) = 0.5$

posterior probability :

$$q_0(x) = P(\text{male}/x) = P(0/x) = \frac{f(x/0)\,P(0)}{f_X(x)} = \frac{P_0 f_0(x)}{f_X(x)}$$

$$q_1(x) = P(\text{female}/x) = P(1/x) = \frac{f(x/1)\,P(1)}{f_X(x)} = \frac{P_1 f_1(x)}{f_X(x)}$$

$$f_X(x) = f_X(x/0)\,P(0) + f_X(x/1)\,P(1)$$
$$= f_0(x)\,P_0 + f_1(x)\,P_1$$

Baye's classifier:

$$h_B(x) = 0 \quad \text{if } q_0(x) > q_1(x)$$
$$= 1 \quad \text{if } q_1(x) > q_0(x)$$

So, the decision boundary we have to calculate:

$\longrightarrow$

$\because$ if $q_0(x) \geq q_1(x) \Rightarrow$ class 0

$$\Rightarrow \frac{P_0 f_0(x)}{f_X(x)} \geq \frac{P_1 f_1(x)}{f_X(x)}$$

$$\Rightarrow P_0 f_0(x) \geq P_1 f_1(x) \quad [P_0 = P_1 = 0.5]$$

$$\Rightarrow f_0(x) \geq f_1(x)$$

$$\Rightarrow \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu_0)^2}{2\sigma^2}} \geq \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma^2}}$$

$$\Rightarrow (x-\mu_0)^2 \leq (x-\mu_1)^2$$

$$\Rightarrow x^2 - 2x\mu_0 + \mu_0^2 \leq x^2 - 2x\mu_1 + \mu_1^2$$

$$\Rightarrow x \leq \frac{\mu_1+\mu_0}{2}$$

$\therefore$ class 0 : $x \leq \frac{\mu_1+\mu_0}{2}$

class 1 : $x > \frac{\mu_1+\mu_0}{2}$

Decision boundary $= \frac{5+5.8}{2} = 5.4$ feet (Ans)

<u>misclassification errors:</u>

$$\text{errors} = P(e) = P(R_1, 0) + P(R_0, 1)$$
$$= P_0 P(R_1/0) + P_1 P(R_0/1)$$
$$= P_0 \int_{-\infty}^{5.4} f_0(x)\,dx + P_1 \int_{5.4}^{\infty} f_1(x)\,dx$$

By calculating, we get, ~~errors~~ =

$P(e) = 0.3445$ (Ans)

Case-B

prior probabilities :

$P_0 = P(\text{class} = \text{male}) = 0.1$

$P_1 = P(\text{class} = \text{female}) = 0.9$

posterior probability:

$$q_0(x) = \frac{P_0 f_0(x)}{f_X(x)}$$

$$q_1(x) = \frac{P_1 f_1(x)}{f_X(x)}$$

Similarly, $f_X(x) = P_0 f_0(x) + P_1 f_1(x)$

Decision boundary:

if $q_1(x) > q_0(x) \Rightarrow$ class 1

$$\Rightarrow \frac{P_1 f_1(x)}{f_X(x)} > \frac{P_0 f_0(x)}{f_X(x)}$$

$$\Rightarrow P_1 \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma^2}} > P_0 \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu_0)^2}{2\sigma^2}}$$

$$\Rightarrow e^{-\frac{(x-\mu_1)^2}{2\sigma^2}} > \frac{P_0}{P_1} e^{-\frac{(x-\mu_0)^2}{2\sigma^2}}$$

$$\Rightarrow -\frac{(x-\mu_1)^2}{2\sigma^2} > \ln\left(\frac{P_0}{P_1}\right) - \frac{(x-\mu_0)^2}{2\sigma^2}$$

$$\Rightarrow (x-\mu_1)^2 < 2\sigma^2 \ln\left(\frac{P_1}{P_0}\right) + (x-\mu_0)^2$$

$$\Rightarrow x^2 - 2x\mu_1 + \mu_1^2 < 2\sigma^2 \ln\left(\frac{P_1}{P_0}\right) + x^2 - 2x\mu_0 + \mu_0^2$$

$$\Rightarrow 2\sigma^2 \ln\left(P_1/P_0\right) + 2x(\mu_1 - \mu_0) > (\mu_1^2 - \mu_0^2)$$

$$\Rightarrow x > \frac{\mu_1^2 - \mu_0^2}{2(\mu_1 - \mu_0)} - \frac{\sigma^2}{(\mu_1 - \mu_0)} \ln\left(P_1/P_0\right)$$

Class 0 : $x \leq \left(\frac{\mu_1+\mu_0}{2}\right) - \frac{\sigma^2}{(\mu_1-\mu_0)} \ln(P_1/P_0)$

Class 1 : $x > \left(\frac{\mu_1+\mu_0}{2}\right) - \frac{\sigma^2}{\mu_1-\mu_0} \ln(P_1/P_0)$

Decision boundary $= \frac{5+5.8}{2} - \frac{1}{5-5.8} \ln\left(\frac{0.9}{0.1}\right)$

$= 8.146$ feet

<u>misclassification error:</u>

$$\text{error} = P(e) = P_0 \int_{-\infty}^{8.146} f_0(x)\,dx + P_1 \int_{8.146}^{\infty} f_1(x)\,dx$$

By calculating, we get

$P(e) = 0.099$ (Ans)