Report

```
from google.colab import drive
drive.mount('/content/drive')
```

Importing corpus from my drive.

```
[8]  folder_zero = "/content/drive/MyDrive/Corpus_Learn_Handwritten_Characters/Zero"

     from PIL import Image
     import numpy as np
     import matplotlib.pyplot as plt


     import os
     from os import listdir

     from array import *

     totalnumberofzeroes = 0
     totalnumberoftruezeroes = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])
     probabilityoftrueomegaforzero = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])


     for images in os.listdir(folder_zero):
       totalnumberofzeroes += 1
       zeroimage = Image.open(folder_zero+"/"+images)
       zeroimage = zeroimage.convert('1')
       zeroimage = zeroimage.resize((4,4))
       zeroimagebinary = np.array(zeroimage)
       for i in range(4):
         for j in range(4):
           if zeroimagebinary[i][j] == True:
             n = (4*i)+j
             totalnumberoftruezeroes[n]+=1
     print("totalzeroes: "+str(totalnumberofzeroes))
     for i in range(16):
       probabilityoftrueomegaforzero[i] = totalnumberoftruezeroes[i]/totalnumberofzeroes
       print("ω("+str(i+1)+")"+"\t"+str(probabilityoftrueomegaforzero[i]))
```

```
totalzeroes: 76
ω(1)    0.8552631735801697
ω(2)    0.8552631735801697
ω(3)    0.8815789222717285
ω(4)    0.9078947305679321
ω(5)    0.8815789222717285
ω(6)    0.8947368264198303
ω(7)    0.8157894611358643
ω(8)    0.8947368264198303
ω(9)    0.9210526347160339
ω(10)   0.8157894611358643
ω(11)   0.9078947305679321
ω(12)   0.8421052694320679
ω(13)   0.7763158082962036
ω(14)   0.8815789222717285
ω(15)   0.8421052694320679
ω(16)   0.9078947305679321
```

Convert each image of zero from corpus of zeroes to binary form from since it was in RGB format initially.

Then convert binary image of zero to 4 cross 4 pixels (initially it could be in any pixels).

4 cross 4 pixels of binary image implies 4 cross 4 matrix with zeroes and ones.

w1, w2……w16 correspond to 4 cross 4 matrix entries where each entry corresponds to probability of that pixel being true considering all zeroes from zero corpus.

wi=total number of zeroes with ith pixel being true(1)/total number of zeroes in the corpus.

```
[9]  folder_one = "/content/drive/MyDrive/Corpus_Learn_Handwritten_Characters/One"

     from PIL import Image
     import numpy as np
     import matplotlib.pyplot as plt


     import os
     from os import listdir

     from array import *

     totalnumberofones = 0
     counttrueones = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])
     probabilityoftrueomegaforone = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])

     for images in os.listdir(folder_one):
       totalnumberofones += 1
       oneimage = Image.open(folder_one+"/"+images)
       oneimage = oneimage.convert('1')
       oneimage = oneimage.resize((4,4))
       binaryimageofone = np.array(oneimage)
       for i in range(4):
         for j in range(4):
           if binaryimageofone[i][j] == True:
             n = (4*i)+j
             counttrueones[n]+=1
     print("total ones: "+str(totalnumberofones))
     for i in range(16):
       probabilityoftrueomegaforone[i] = counttrueones[i]/totalnumberofones
       print("ω("+str(i+1)+")"+"\t"+str(probabilityoftrueomegaforone[i]))
```

```
total ones: 75
ω(1)    0.8666666746139526
ω(2)    0.8533333539962769
ω(3)    0.8533333539962769
ω(4)    0.8399999737739563
ω(5)    0.800000011920929
ω(6)    0.8533333539962769
ω(7)    0.8533333539962769
ω(8)    0.800000011920929
ω(9)    0.8533333539962769
ω(10)   0.8399999737739563
ω(11)   0.8266666531562805
ω(12)   0.8666666746139526
ω(13)   0.800000011920929
ω(14)   0.8266666531562805
ω(15)   0.8533333539962769
ω(16)   0.8133333325386047
```

Convert each image of one from corpus of ones to binary form from since it was in RGB format initially.

Then convert binary image of one to 4 cross 4 pixels (initially it could be in any pixels).

4 cross 4 pixels of binary image implies 4 cross 4 matrix with zeroes and ones.

w1, w2……w16 correspond to 4 cross 4 matrix entries where each entry corresponds to probability of that pixel being true considering all ones from one corpus.

wi=total number of ones with ith pixel being true(1)/total number of ones in the corpus.

```
[10] totalcount=totalnumberofzeroes+totalnumberofones

     probabilityofzero = totalnumberofzeroes/totalcount
     probabilityofone = totalnumberofones/totalcount

     normalizedprobabilityoftrueomegaforzero = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])
     normalizedprobabilityoftrueomegaforone = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])

     for i in range(16):
       normalizedprobabilityoftrueomegaforzero[i] = probabilityoftrueomegaforzero[i]*probabilityofzero

     for i in range(16):
       normalizedprobabilityoftrueomegaforone[i] = probabilityoftrueomegaforone[i]*probabilityofone


     wcorrespondstoimagezero = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])
     wcorrespondstoimageone = array('f',[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0])

     for i in range(16):
       if(normalizedprobabilityoftrueomegaforzero[i]<=normalizedprobabilityoftrueomegaforone[i]):
         wcorrespondstoimageone[i]=1
       else:
         wcorrespondstoimagezero[i]=1
```

Just normalising the probabilities (w0, w1, w2,…..w16) since corpus has different number of zeroes and ones. Therefore multiply each probability(wi) with suitable number as shown above.

```
[ ]  inputimage = Image.open("/content/drive/MyDrive/zero22.png")
     inputimage = inputimage.convert('1')
     inputimage = inputimage.resize((4,4))
     inputimage_binary = np.array(inputimage)

     countforimagetobezero = 0
     countforimagetobeone = 0

     for i in range(4):
       for j in range(4):
         if(inputimage_binary[i][j]==wcorrespondstoimagezero[4*i+j]):
           countforimagetobezero+=1
         else:
           countforimagetobeone+=1

     totalcount = countforimagetobezero + countforimagetobeone

     probabilityofimagetobezero = countforimagetobezero/totalcount
     probabilityofimagetobeone = countforimagetobeone/totalcount


     print("Probability of image to be zero is: "+str(probabilityofimagetobezero))
     print("Probability of image to be one is: "+str(probabilityofimagetobeone))
```

```
Probability of image to be zero is: 0.625
Probability of image to be one is: 0.375
```

An input image of zero is taken and converted to binary form from since it was in RGB format initially.

Then converted that image of zero to 4 cross 4 pixels (initially it could be in any pixels).

4 cross 4 pixels of binary image of zero implies 4 cross 4 matrix with entries as zeroes and ones.

For calculating probability

Example:

Let's say w1 for zero corpus =0.6

w1 for one corpus =0.3

For input image w1 could be either 0 or 1.

If w1 of input image=0

Compare w1 of input image with w1 for zero corpus and w1 for one's corpus.

Take the smaller value amongst them.

If w1 for zero gets selected then image is biased towards 0 for that pixel.

If w1 for one gets selected then image is biased towards 1 for that pixel.

Similarly calculate the biasness for all pixels that is for all wi and then calculate the probability.

Probability of image to be one=total no. of pixels biased towards image one/total number of pixels

Probability of image to be zero=total no. of pixels biased towards image zero/total number of pixels