```
import numpy as np
import random
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
```

```
"""
a) Generate 100 random coordinate points (2D data representing height (h) and weight (w)) and label th
"""
kids = np.random.randint([1, 20, 0], [20, 150, 1], size=(100, 3))

"""
b) Follow exact same procedure for generating 100 data for another class: ‘adult’.
Make sure that they are not completely separated.
"""
adults = np.random.randint([10, 100, 1], [60, 300, 2], size=(100, 3))
print('kids { ')
print(kids[:3])
print('}')
print('.....................')
print('Adults { ')
print(adults[:3])
print('}')
```

```
kids {
[[  7  52   0]
 [  9 111   0]
 [ 13  26   0]]
}
.....................
Adults {
[[ 26 259   1]
 [ 37 112   1]
 [ 17 272   1]]
}
```

```
df = np.concatenate((kids, adults))
df[97:103]
```

```
array([[ 14,  96,   0],
       [ 13, 125,   0],
       [ 18, 129,   0],
       [ 26, 259,   1],
       [ 37, 112,   1],
       [ 17, 272,   1]])
```

```
classifier = df[:,-1]
data = df[:,0:2]
print(classifier)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

```
"""
c) Train 80% of them using Gaussian classify and compute their prior probabilities (for both classes)
d) Test the models using the remaining 20% of the data and check whether you build your model right.
   (hint: posterior probability can help you compute this).
"""
X_train, X_test, y_train, y_test = train_test_split( data , classifier , test_size = 0.20)

```

```
modelG = GaussianNB()
modelG.fit(X_train, y_train)
```

```
GaussianNB()
```

```
print("Trained Data: ",modelG.score(X_train,y_train))
print("Test Data Result: ", modelG.score(X_test,y_test))
```

```
Trained Data:  0.96875
Test Data Result:  0.95
```

```
"""
(hint: posterior probability can help you compute this).
"""
probability_Kids = sum(y_train)
total = len(y_train)

first_Kids = probability_Kids / total
first_Adults = 1 - first_Kids

print(first_Kids)
print(first_Adults)
```

```
0.50625
0.49375
```

```
model = GaussianNB(priors=[first_Kids, first_Adults])

model.fit(X_train, y_train)
print(model.score(X_test, y_test))
```

```
0.95
```

✓ 0s completed at 10:39 PM