Practical 6

```
# importing libraries
from sklearn.datasets import load_iris
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics

# loading data
iris = load_iris()
X = iris.data
y = iris.target

# splitting X and y into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.4, random_state=1)

# training the model on training set
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# making predictions on the testing set
y_pred = gnb.predict(X_test)

# comparing actual response values (y_test) with predicted response
values (y_pred)
print("Gaussian Naive Bayes model accuracy(in %):",
metrics.accuracy_score(y_test, y_pred)*100)
print("confusion matrix is\n",confusion_matrix(y_test,y_pred))
```

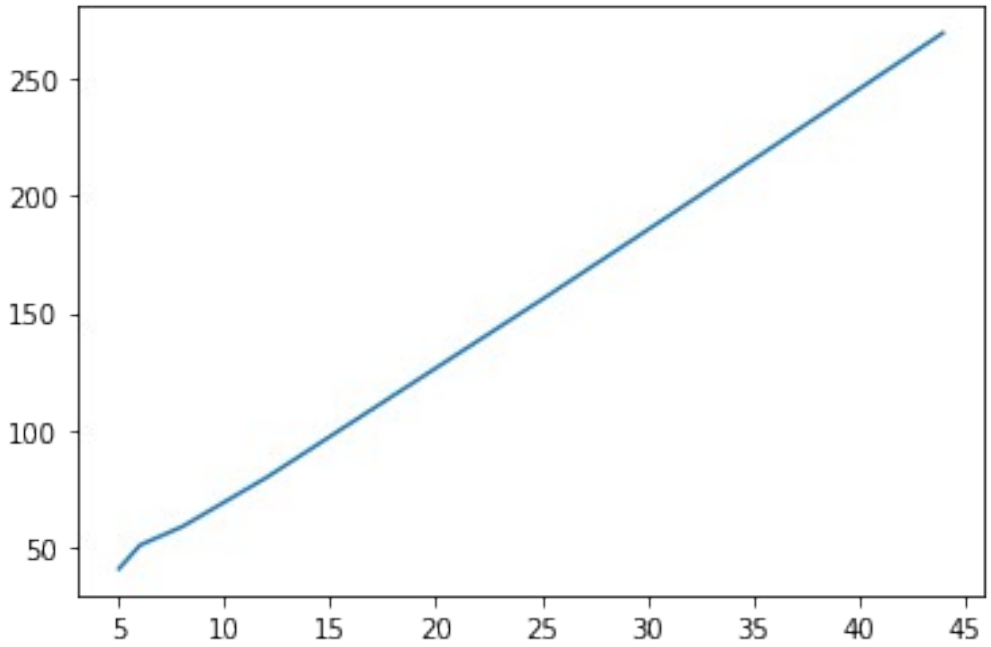
```
Gaussian Naive Bayes model accuracy(in %): 95.0
confusion matrix is
 [[19  0  0]
 [ 0 19  2]
 [ 0  1 19]]
```

Practical 7

```
import matplotlib.pyplot as plt

experience = [3, 8, 9, 13, 3, 6, 11, 21, 1, 16]
salary = [30, 57, 64, 72, 36, 43, 59, 90, 20, 83]
sumExp = 0
sumSal = 0
numerator = 0
denominator = 0
yRequired = 0
xAxis = []
yAxis = []
```

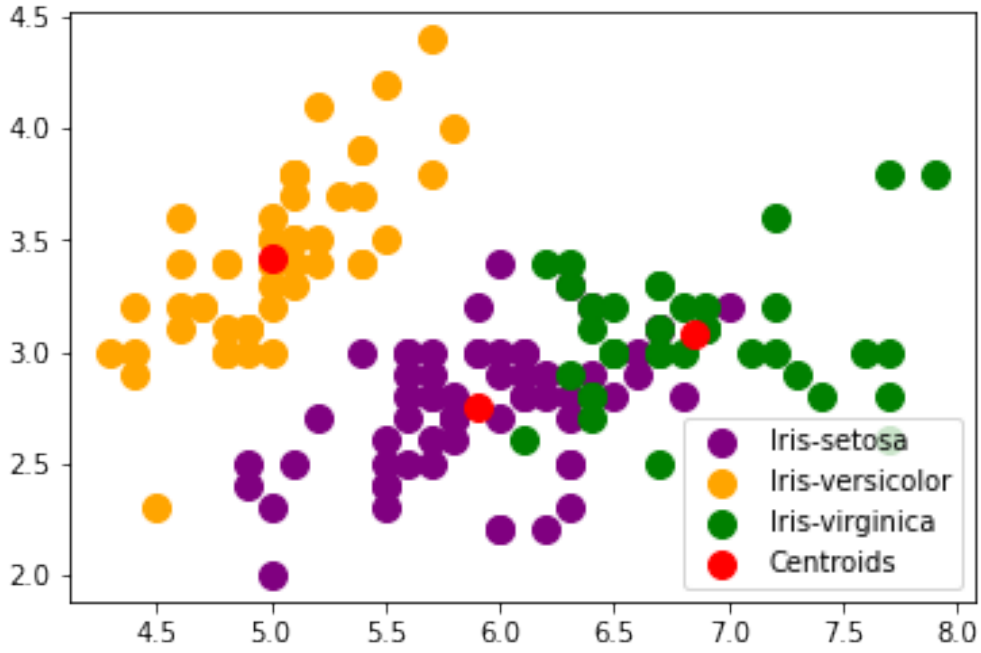
```
for j in range(10):
    userInput = int(input('Enter experience in years: '))
    xAxis.append(userInput)
    n = len(experience)

    for i in range(n):
        sumExp = sumExp + experience[i]
        sumSal = sumSal + salary[i]

    xBar = sumExp / n
    yBar = sumSal / n

    for i in range(n):
        numerator = numerator + ((experience[i] - xBar) * (salary[i] -
yBar))
        denominator = denominator + ((experience[i] - xBar) *
(experience[i] - xBar))

    w1 = numerator / denominator
    w0 = yBar - (w1 * xBar)

    yRequired = w0 + (w1 * userInput)

    yAxis.append(yRequired)

print("Salary based on given experiences: ")
print(yAxis)

plt.plot(xAxis, yAxis)

plt.show()
```

```
Enter experience in years: 5
Enter experience in years: 6
Enter experience in years: 8
Enter experience in years: 12
Enter experience in years: 25
Enter experience in years: 29
Enter experience in years: 31
Enter experience in years: 36
Enter experience in years: 40
Enter experience in years: 44
Salary based on given experiences:
[40.896349958205626, 50.97509541367487, 58.861784098750235,
79.9118274618159, 155.72003754106052, 179.49706597007471,
191.41598496970556, 221.4330820536474, 245.52144456545406,
269.66370684168237]
```



Practical 9

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

iris = pd.read_csv(r'/content/Iris.csv')
x = iris.iloc[:,[1,2,3,4]].values

iris_setosa = iris.loc[iris["Species"] == "Iris-setosa"]
iris_virginica = iris.loc[iris["Species"] == "Iris-virginica"]
iris_versicolor = iris.loc[iris["Species"] == "Iris-versicolor"]

kmeans = KMeans(n_clusters = 3, init = 'k-means++', max_iter = 300,
n_init = 10, random_state = 0)
kmeans.fit(x)

y_kmeans = kmeans.fit_predict(x)

plt.scatter(x[y_kmeans==0,0],x[y_kmeans==0,1], s=100, c='purple',
label = 'Iris-setosa')
plt.scatter(x[y_kmeans==1,0],x[y_kmeans==1,1], s=100, c='orange',
label = 'Iris-versicolor')
plt.scatter(x[y_kmeans==2,0],x[y_kmeans==2,1], s=100, c='green', label
= 'Iris-virginica')
plt.scatter(kmeans.cluster_centers_[:,0],
kmeans.cluster_centers_[:,1], s=100, c='red',label='Centroids')
plt.legend();
```


4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
Iris-setosa
Iris-versicolor
Iris-virginica
Centroids