Lab 7

Q Write a Python program to implement Bayes' Theorem for reasoning under uncertainty using a medical test example.

Assume that 1% of people have a disease, the test correctly detects the disease 99% of the time, and gives a false positive 2% of the time.

Calculate the posterior probability that a person actually has the disease given a positive test result.

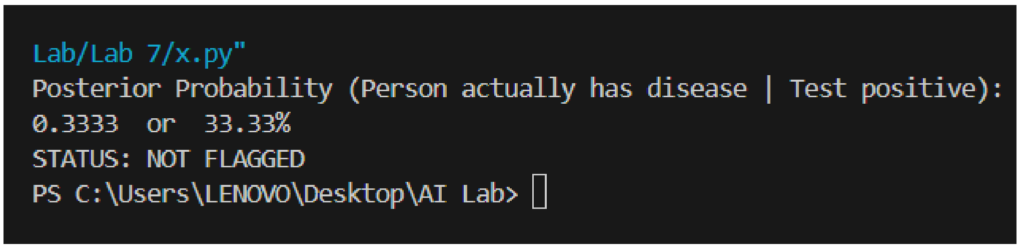
Display the calculated probability and interpret the result by printing whether the person should be

FLAGGED (if ≥ 0.5) or NOT FLAGGED (if < 0.5).

Code:-

```
# Bayes Theorem for Medical Test Example


# Given probabilities

P_disease = 0.01            # Prevalence: 1% have the disease

P_positive_given_disease = 0.99  # Test detects disease correctly (true positive rate)

P_positive_given_no_disease = 0.02  # False positive rate: 2%


# Step 1: Probability of NOT having disease

P_no_disease = 1 - P_disease


# Step 2: Total probability of testing positive

P_positive = (P_positive_given_disease * P_disease) + \

             (P_positive_given_no_disease * P_no_disease)


# Step 3: Bayes Theorem (Posterior Probability)

P_disease_given_positive = (P_positive_given_disease * P_disease) / P_positive


# Step 4: Print result
```

```
print("Posterior Probability (Person actually has disease | Test positive):")
print(f"{P_disease_given_positive:.4f}  or  {P_disease_given_positive*100:.2f}%")


# Step 5: Flagging logic
if P_disease_given_positive >= 0.5:
    print("STATUS: FLAGGED")
else:
    print("STATUS: NOT FLAGGED")
```



```
Lab/Lab 7/x.py"
Posterior Probability (Person actually has disease | Test positive):
0.3333  or  33.33%
STATUS: NOT FLAGGED
PS C:\Users\LENOVO\Desktop\AI Lab> 
```

Lab 8

Q Write a Python program to implement Linear Regression using a simple dataset of input (X) and output (Y) values.

Dataset (use exactly this):

X = [1, 2, 3, 4, 5, 6, 7] (Years of Experience)

Y = [40, 45, 50, 55, 60, 65, 70] (Salary in ₹000)

Your program should calculate the best-fit line using the formula Y=mX+cY = mX + cY=mX+c, where m and c are obtained from the data.

Use the scikit-learn or manual calculation method to train the model, predict values, and plot both the data points and the regression line on a graph using matplotlib.

Display the slope, intercept, and predicted outputs clearly along with the plotted graph.

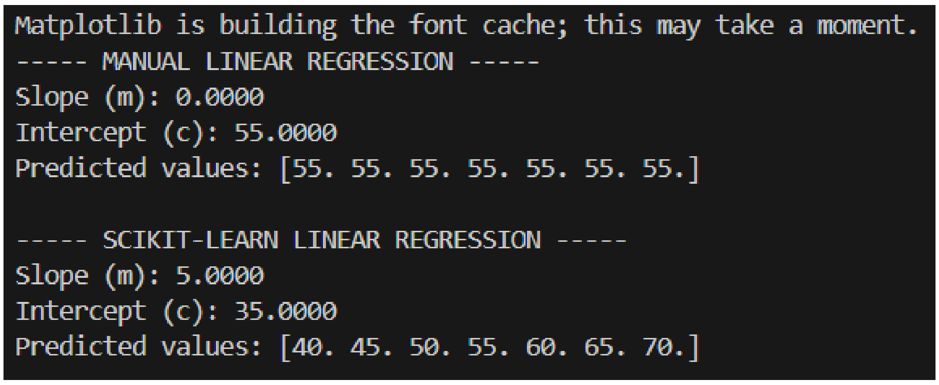
Code:-

```
import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear_model import LinearRegression


# Dataset

X = np.array([1, 2, 3, 4, 5, 6, 7]).reshape(-1, 1)

Y = np.array([40, 45, 50, 55, 60, 65, 70])


# Mean of X and Y

mean_x = np.mean(X)

mean_y = np.mean(Y)


# Calculate slope m

numerator = np.sum((X - mean_x) * (Y - mean_y))

denominator = np.sum((X - mean_x)**2)

m_manual = numerator / denominator


# Calculate intercept c
```

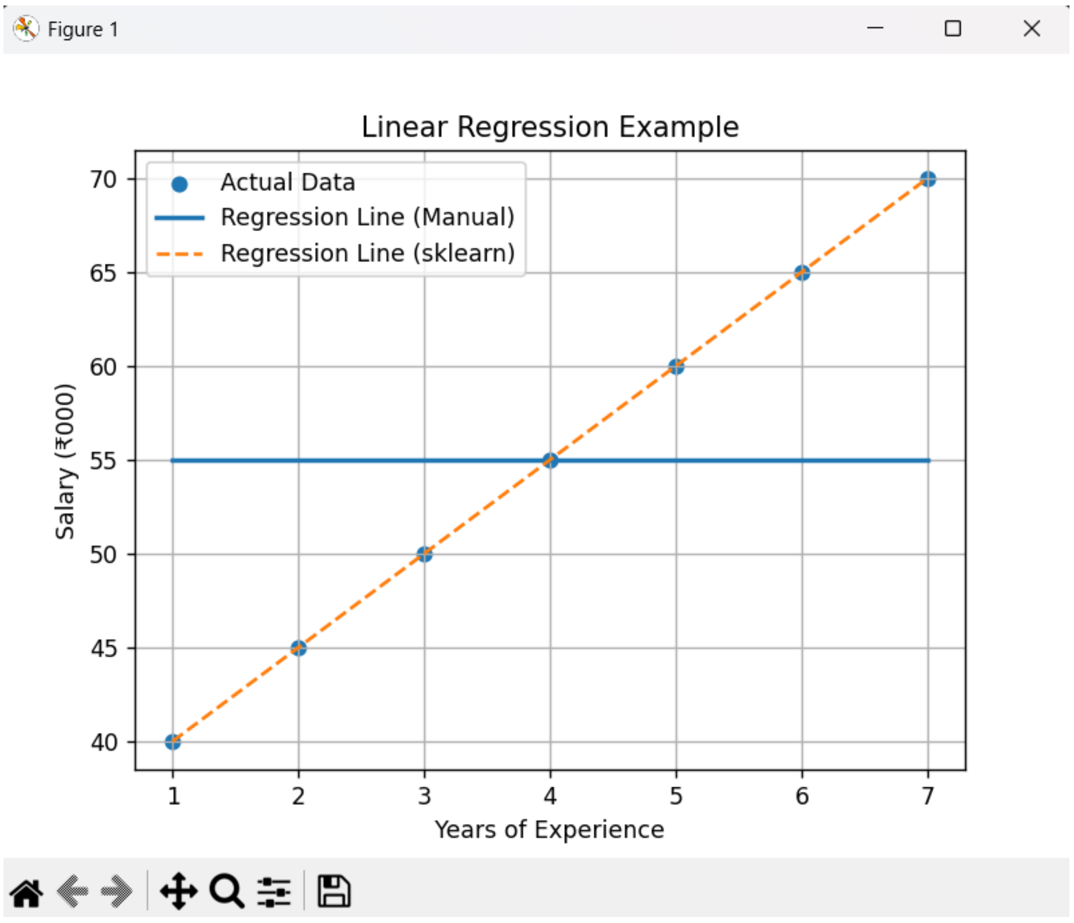
```
c_manual = mean_y - m_manual * mean_x

# Predict using manual model
Y_pred_manual = m_manual * X + c_manual

print("----- MANUAL LINEAR REGRESSION -----")
print(f"Slope (m): {m_manual:.4f}")
print(f"Intercept (c): {c_manual:.4f}")
print("Predicted values:", Y_pred_manual.flatten())
print()

model = LinearRegression()
model.fit(X, Y)
Y_pred_sklearn = model.predict(X)

print("----- SCIKIT-LEARN LINEAR REGRESSION -----")
print(f"Slope (m): {model.coef_[0]:.4f}")
print(f"Intercept (c): {model.intercept_:.4f}")
print("Predicted values:", Y_pred_sklearn)
print()

plt.scatter(X, Y, label="Actual Data", marker='o')
plt.plot(X, Y_pred_manual, label="Regression Line (Manual)", linewidth=2)
plt.plot(X, Y_pred_sklearn, linestyle='dashed', label="Regression Line (sklearn)")

plt.xlabel("Years of Experience")
plt.ylabel("Salary (₹000)")
plt.title("Linear Regression Example")
plt.legend()
```

plt.grid(True)

plt.show()



```
Matplotlib is building the font cache; this may take a moment.
----- MANUAL LINEAR REGRESSION -----
Slope (m): 0.0000
Intercept (c): 55.0000
Predicted values: [55. 55. 55. 55. 55. 55. 55.]

----- SCIKIT-LEARN LINEAR REGRESSION -----
Slope (m): 5.0000
Intercept (c): 35.0000
Predicted values: [40. 45. 50. 55. 60. 65. 70.]
```



Lab 9

Q Write a Python program to perform a performance analysis on the chosen dataset using at least one machine learning algorithm (e.g., Linear Regression, Decision Tree, Logistic Regression, or KNN).

Split the data into training and testing sets, train the model, and calculate performance metrics such as accuracy, precision, recall, F1-score, or R2 score depending on the algorithm used.

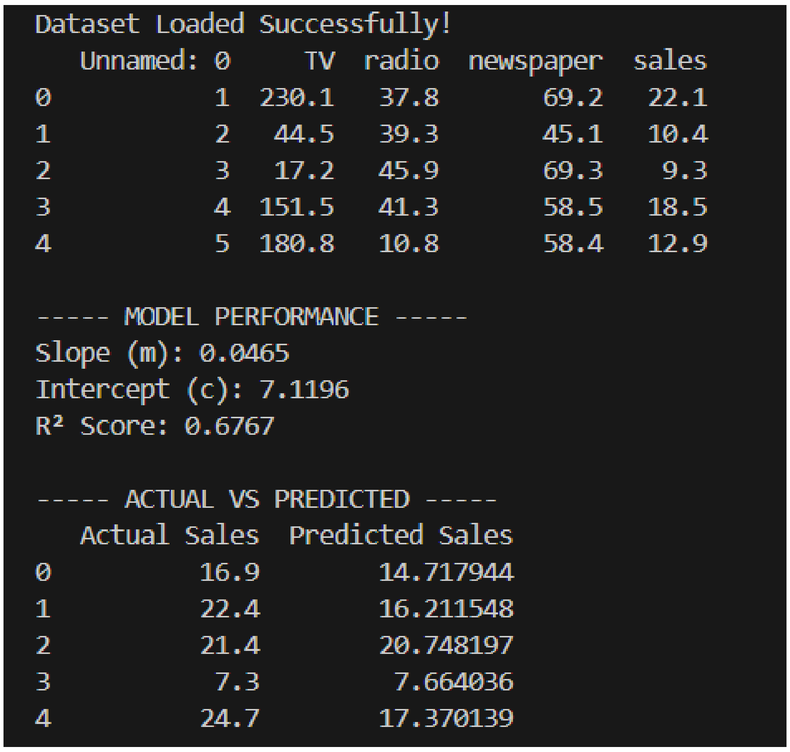
Finally, display and interpret the results, comparing actual vs predicted values, and plot a suitable graph (like confusion matrix or regression line) for visual understanding.

Dataset:

Use any real-world dataset of your choice from Kaggle or github.

Code:-

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import r2_score


url = "https://raw.githubusercontent.com/selva86/datasets/master/Advertising.csv"

data = pd.read_csv(url)


print("Dataset Loaded Successfully!")

print(data.head())

print()


X = data[['TV']]   # Independent variable

Y = data['sales']  # Dependent variable


X_train, X_test, Y_train, Y_test = train_test_split(

   X, Y, test_size=0.2, random_state=42
)

model = LinearRegression()
model.fit(X_train, Y_train)

Y_pred = model.predict(X_test)
r2 = r2_score(Y_test, Y_pred)

print("----- MODEL PERFORMANCE -----")
print(f"Slope (m): {model.coef_[0]:.4f}")
print(f"Intercept (c): {model.intercept_:.4f}")
print(f"R² Score: {r2:.4f}")
print()

comparison = pd.DataFrame({
    "Actual Sales": Y_test.values,
    "Predicted Sales": Y_pred
})
print("----- ACTUAL VS PREDICTED -----")
print(comparison.head())
print()
plt.scatter(X_test, Y_test, label="Actual Data", marker='o')
plt.plot(X_test, Y_pred, label="Predicted Regression Line", linewidth=2)

plt.xlabel("TV Advertising Budget")
plt.ylabel("Sales")
plt.title("Linear Regression - Actual vs Predicted Sales")
plt.legend()
plt.grid(True)
plt.show()
```



```
Dataset Loaded Successfully!
   Unnamed: 0     TV  radio  newspaper  sales
0           1  230.1   37.8       69.2   22.1
1           2   44.5   39.3       45.1   10.4
2           3   17.2   45.9       69.3    9.3
3           4  151.5   41.3       58.5   18.5
4           5  180.8   10.8       58.4   12.9

----- MODEL PERFORMANCE -----
Slope (m): 0.0465
Intercept (c): 7.1196
R² Score: 0.6767

----- ACTUAL VS PREDICTED -----
   Actual Sales  Predicted Sales
0          16.9        14.717944
1          22.4        16.211548
2          21.4        20.748197
3           7.3         7.664036
4          24.7        17.370139
```

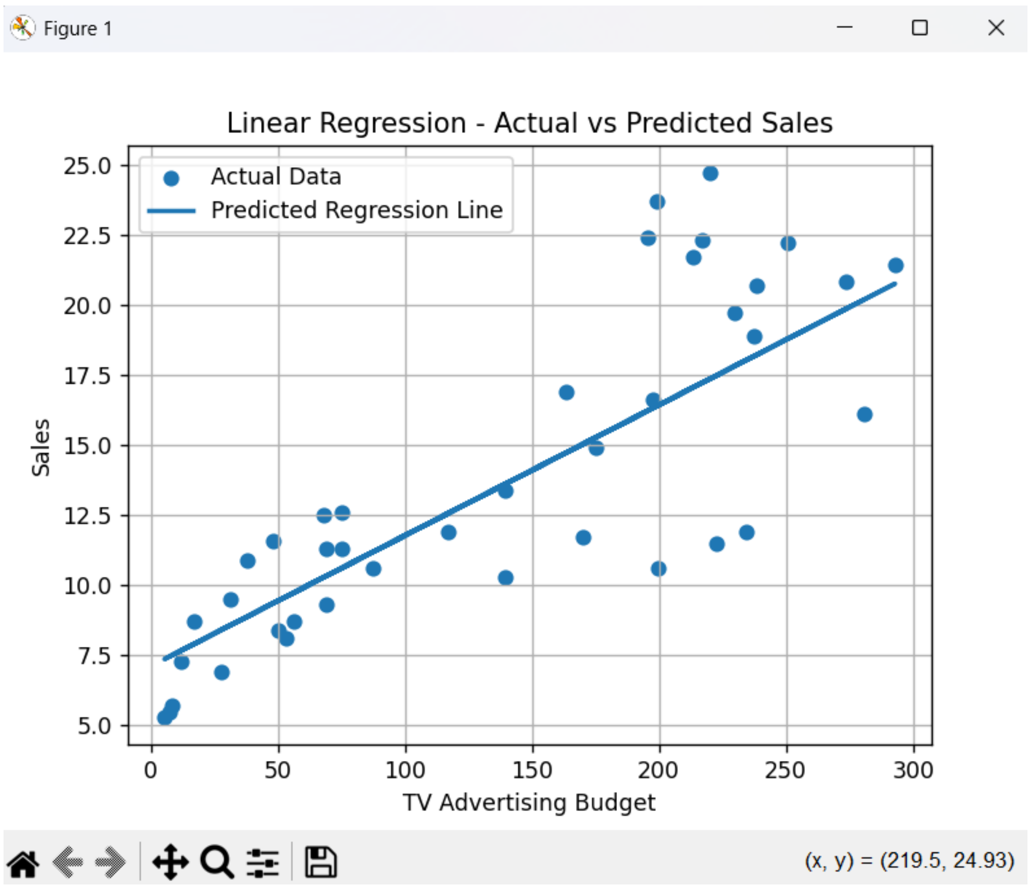


Figure 1
Linear Regression - Actual vs Predicted Sales
Actual Data
Predicted Regression Line
Sales
TV Advertising Budget
(x, y) = (219.5, 24.93)