**PROGRAM 9 :** Develop a program to implement the Naive Bayesian classifier considering Olivetti Face Data set for training. Compute the accuracy of the classifier, considering a few test data sets.

import numpy as np

import joblib

from sklearn.model\_selection import train\_test\_split, cross\_val\_score

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

import matplotlib.pyplot as plt

# Load dataset from local file

data = joblib.load('olivetti\_faces.pkl')

X = data.data

y = data.target

# Split into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Train Naive Bayes classifier

gnb = GaussianNB()

gnb.fit(X\_train, y\_train)

# Predict and evaluate

y\_pred = gnb.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Accuracy: {accuracy \* 100:.2f}%')

print("\nClassification Report:")

print(classification\_report(y\_test, y\_pred, zero\_division=1))

print("\nConfusion Matrix:")

print(confusion\_matrix(y\_test, y\_pred))

# Cross-validation

cross\_val\_accuracy = cross\_val\_score(gnb, X, y, cv=5, scoring='accuracy')

print(f'\nCross-validation accuracy: {cross\_val\_accuracy.mean() \* 100:.2f}%')

# Plot some predictions

fig, axes = plt.subplots(3, 5, figsize=(12, 8))

for ax, image, label, prediction in zip(axes.ravel(), X\_test, y\_test, y\_pred):

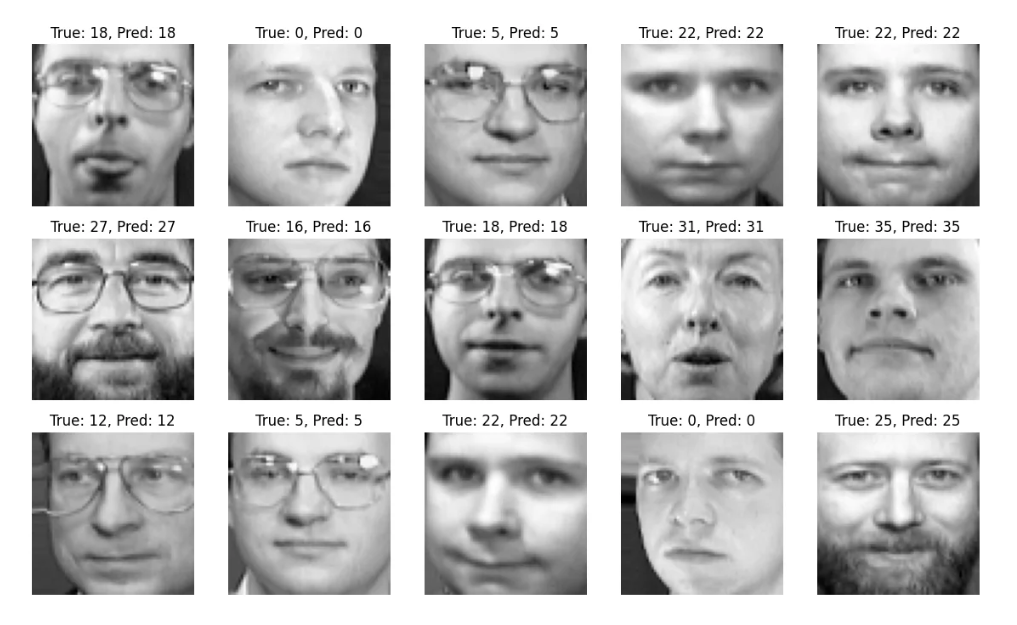
    ax.imshow(image.reshape(64, 64), cmap=plt.cm.gray)

    ax.set\_title(f"True: {label}, Pred: {prediction}")

    ax.axis('off')

plt.tight\_layout()

plt.show()



Accuracy: 80.83%Classification Report:precision recall f1-score support0 0.67 1.00 0.80 2

1 1.00 1.00 1.00 2

2 0.33 0.67 0.44 3

3 1.00 0.00 0.00 5

4 1.00 0.50 0.67 4

5 1.00 1.00 1.00 2

7 1.00 0.75 0.86 4

8 1.00 0.67 0.80 3

9 1.00 0.75 0.86 4

10 1.00 1.00 1.00 3

11 1.00 1.00 1.00 1

12 0.40 1.00 0.57 4

13 1.00 0.80 0.89 5

14 1.00 0.40 0.57 5

15 0.67 1.00 0.80 2

16 1.00 0.67 0.80 3

17 1.00 1.00 1.00 3

18 1.00 1.00 1.00 3

19 0.67 1.00 0.80 2

20 1.00 1.00 1.00 3

21 1.00 0.67 0.80 3

22 1.00 0.60 0.75 5

23 1.00 0.75 0.86 4

24 1.00 1.00 1.00 3

25 1.00 0.75 0.86 4

26 1.00 1.00 1.00 2

27 1.00 1.00 1.00 5

28 0.50 1.00 0.67 2

29 1.00 1.00 1.00 2

30 1.00 1.00 1.00 2

31 1.00 0.75 0.86 4

32 1.00 1.00 1.00 2

34 0.25 1.00 0.40 1

35 1.00 1.00 1.00 5

36 1.00 1.00 1.00 3

37 1.00 1.00 1.00 1

38 1.00 0.75 0.86 4

39 0.50 1.00 0.67 5accuracy 0.81 120

macro avg 0.89 0.85 0.83 120

weighted avg 0.91 0.81 0.81 120Confusion Matrix:

[[2 0 0 ... 0 0 0]

[0 2 0 ... 0 0 0]

[0 0 2 ... 0 0 1]

...

[0 0 0 ... 1 0 0]

[0 0 0 ... 0 3 0]

[0 0 0 ... 0 0 5]]Cross-validation accuracy: 87.25%