

**EXPT NO: 6            A python program to implement face recognition**

**DATE: 24/10/2024            using Support Vector Machine.**

**AIM:**

To write a python program to implement face recognition using

**SVM. PROCEDURE:**

Implementing face recognition using svm involves the following

steps: **Step 1: Import Necessary Libraries**

First, import the libraries that are essential for data manipulation, visualization, and model building.

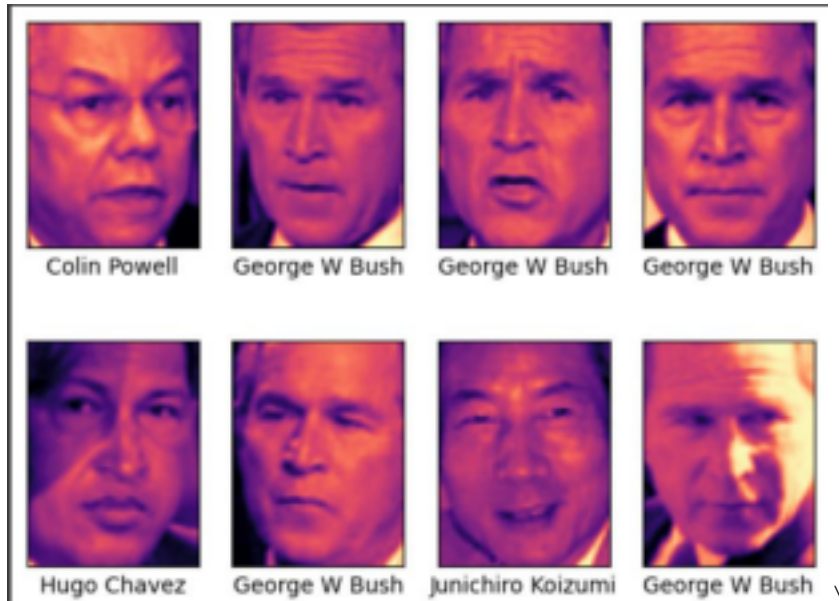
```
from sklearn.datasets import fetch_lfw_people
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.pipeline import make_pipeline
from sklearn.decomposition import PCA as RandomizedPCA
from sklearn.metrics import accuracy_score
```

**Step 2: Load the Dataset**

The dataset can be loaded and display the first few faces of the dataset.

```
faces = fetch_lfw_people(min_faces_per_person=60)
fig, splts = plt.subplots(2, 4)
for i, splts in enumerate(splts.flat):
    splts.imshow(faces.images[i], cmap='magma')
    splts.set(xticks=[], yticks=[],
              xlabel=faces.target_names[faces.target[i]])
```

## OUTPUT:



### Step 4: Split the Data

Split the data into training and testing sets.

Fit the dataset to the model.

```
X = faces.data
y = faces.target
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,
random_state=42)
```

### Step 5:

#### Dimensionality Reduction

Reduce the dimension using Principal Component Analysis

(PCA) Fit the model with SVC.

```
pca = RandomizedPCA(n_components=150, whiten=True,
random_state=42) svc = SVC(kernel='rbf', class_weight='balanced')
model = make_pipeline(pca, svc)
model.fit(X_train, y_train)
```

## Step 6: Make Predictions

Use the model to make predictions on the test data.

```
predictions = model.predict(X_test)
```

## Step 7: Evaluate the Model

Evaluate the model performance using metrics like Accuracy Score and confusion\_matrix

### OUTPUT :

```
predictions = model.predict(X_test)
accuracy = accuracy_score(predictions, y_test)
print(accuracy)
matrix = confusion_matrix(predictions, y_test)
print(matrix)
```

0.8074074074074075

```
[[ 15   1   1   0   0   0   0   0]
 [  4 101   4  20   3   8   1   9]
 [  2   0  39   1   0   0   0   0]
 [  2   4   5 183   5   7   4  10]
 [  0   0   0   1  28   5   0   0]
 [  0   0   0   0   0  13   0   0]
 [  0   0   0   0   0   0  16   0]
 [  0   2   1   0   3   1   0  41]]
```

---

**RESULT:**

This step-by-step process will help us to implement face recognition using SVM and analyzed their performance.