Face Recognition Using SVM

AIM:

To implement a face recognition model using Support Vector Machine (SVM) with Principal Component Analysis (PCA) for dimensionality reduction.

ALGORITHM:

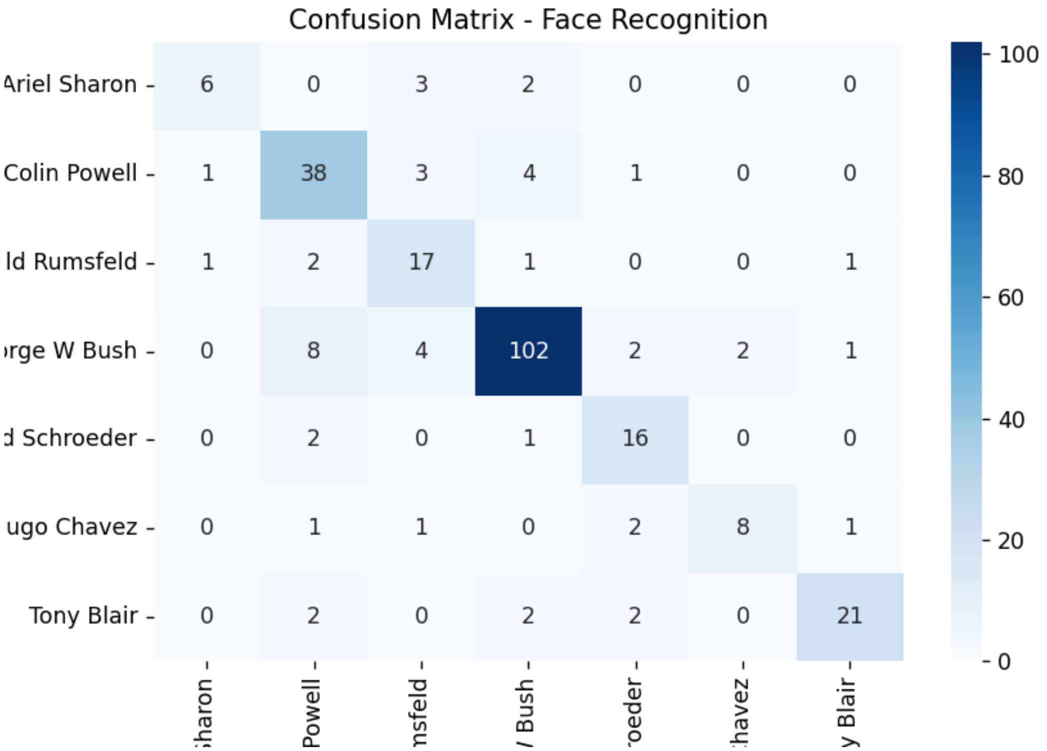
Step 1: Load the Labeled Faces in the Wild (LFW) dataset.
Step 2: Flatten the face images into 1D feature vectors.
Step 3: Normalize the data using StandardScaler.
Step 4: Split the dataset into training and testing sets (80% train, 20% test).
Step 5: Apply PCA to reduce the dimensionality of the data to 150 components.
Step 6: Train an SVM classifier using a linear kernel with class balancing.
Step 7: Predict the labels for the test data using the trained SVM model.
Step 8: Calculate and display the accuracy of the model.
Step 9: Display a confusion matrix to evaluate the model's performance.
Step 10: Test the model with a sample image and show the predicted label.

SOURCE CODE:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import fetch_lfw_people
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix

# Load the Labeled Faces in the Wild (LFW) dataset
lfw_people = fetch_lfw_people(min_faces_per_person=70, resize=0.4)
X = lfw_people.images  # Face images (Gray-scale)
y = lfw_people.target  # Person labels
target_names = lfw_people.target_names  # Names of people

# Flatten images for SVM input (Convert 2D images to 1D feature vectors)
```

```
n_samples, h, w = X.shape
X = X.reshape(n_samples, h * w)

# Normalize data
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Split data (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Apply PCA (Principal Component Analysis) for dimensionality reduction
n_components = 150  # Reduce features to 150 dimensions
pca = PCA(n_components=n_components, whiten=True)
X_train_pca = pca.fit_transform(X_train)
X_test_pca = pca.transform(X_test)

# Train SVM classifier
svm_classifier = SVC(kernel="linear", class_weight="balanced", probability=True)
svm_classifier.fit(X_train_pca, y_train)

# Test the model
y_pred = svm_classifier.predict(X_test_pca)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Face Recognition Model Accuracy: {accuracy * 100:.2f}%")

# Display Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 5))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=target_names,
yticklabels=target_names)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix - Face Recognition")
plt.show()

# Test with a sample image
sample_idx = 5  # Choose any index from test set
plt.imshow(lfw_people.images[sample_idx], cmap="gray")
plt.title(f"Actual: {target_names[y_test[sample_idx]]} \nPredicted:
{target_names[y_pred[sample_idx]]}")
plt.axis("off")
plt.show()
```

OUTPUT:


Confusion Matrix - Face Recognition
Ariel Sharon
Colin Powell
ld Rumsfeld
rge W Bush
d Schroeder
ugo Chavez
Tony Blair
6 0 3 2 0 0 0
1 38 3 4 1 0 0
1 2 17 1 0 0 1
0 8 4 102 2 2 1
0 2 0 1 16 0 0
0 1 1 0 2 8 1
0 2 0 2 2 0 21
haron
Powell
nsfeld
/ Bush
oeder
havez
y Blair
100
80
60
40
20
0

Actual: George W Bush
Predicted: George W Bush

RESULT:

The face recognition model achieved an accuracy of **80.62%**. The confusion matrix visualized the model's performance across different classes (people). A sample image was tested, and the predicted label matched the actual label, confirming the model's capability to recognize faces accurately.