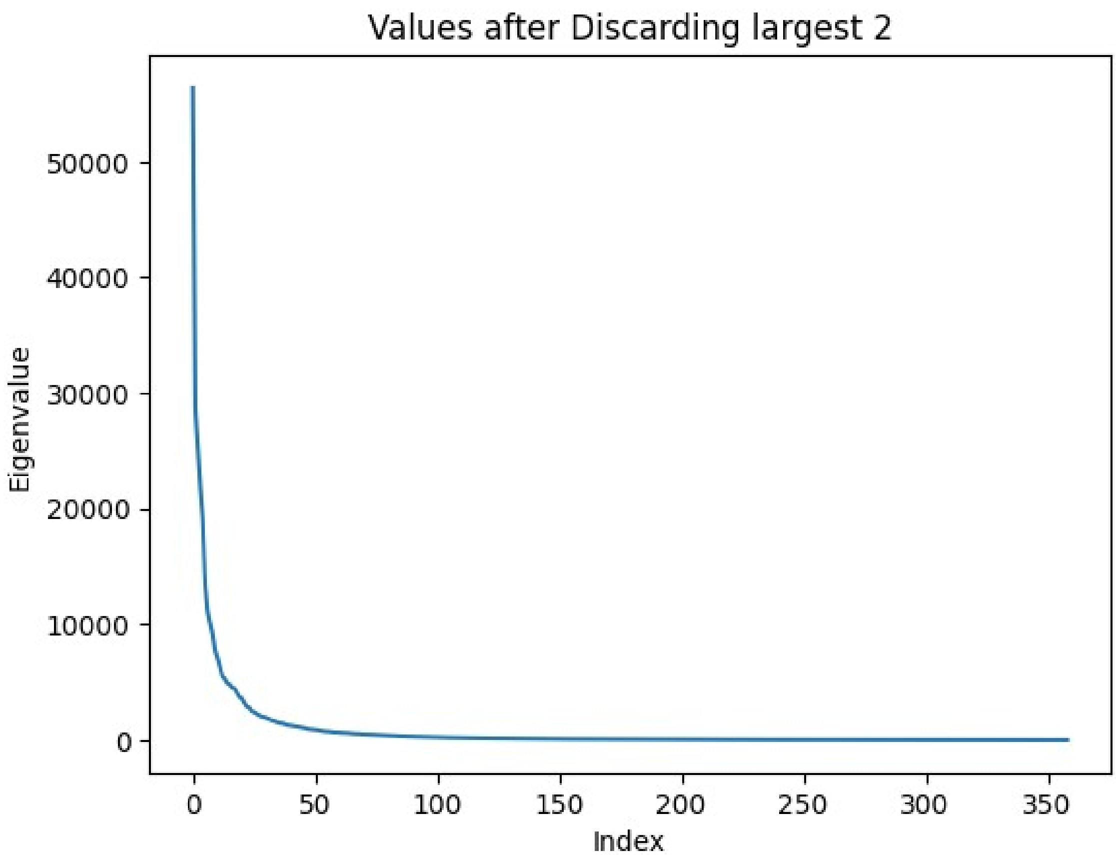
```python
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.io
from scipy.spatial.distance import cdist
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.metrics.cluster import contingency_matrix
from sklearn.metrics import adjusted_rand_score
from scipy.optimize import linear_sum_assignment
import pandas as pd

import os

def load_data(path, data):

    for file in os.listdir(path):
        if file.endswith('.pgm'):
            try:
                valid_path = os.path.join(path, file)
                img = Image.open(valid_path)
                data.append(img)
            except IOError:
                print(f"Ignoring File: {file}")
    return data;

def get_matrix(data):

    matrix = np.array([np.array(elm).flatten() for elm in data])

    matrix = matrix - np.mean(matrix, axis=0)

    return matrix

path = "C:/Users/rique/Downloads/datasets/faces"

faces = []

faces = load_data(path, faces)

face_matrix = get_matrix(faces)
```

a)

```python
# APPLYING PCA
first_pca = PCA()
first_pca.fit(face_matrix)
```

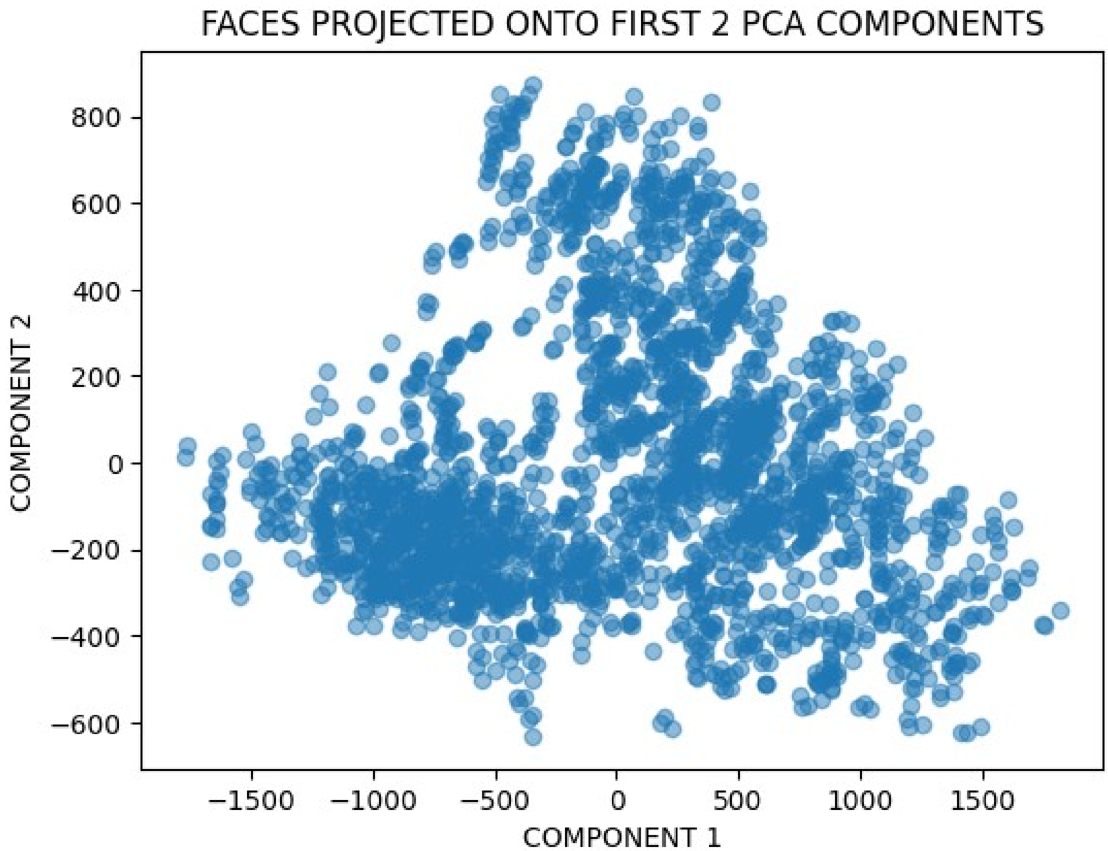
```python
# EIGENVALUES AND DISCARD FIRST TWO
eigenvalues = first_pca.explained_variance_[2:]

# SORTING EIGENVALUES IN DECREASING ORDER
sorted_eigenvalues = np.sort(eigenvalues)[::-1]

# PLOTTING EIGENVALUES
plt.plot(sorted_eigenvalues)
plt.title('Values after Discarding largest 2')
plt.xlabel('Index')
plt.ylabel('Eigenvalue')
plt.show()
```



b)

```python
# APPLYING PCA
second_pca = PCA(n_components=2)
projected = second_pca.fit_transform(face_matrix)
# PLOT
plt.scatter(projected[:, 0], projected[:, 1], alpha=0.5)
plt.xlabel('COMPONENT 1')
```

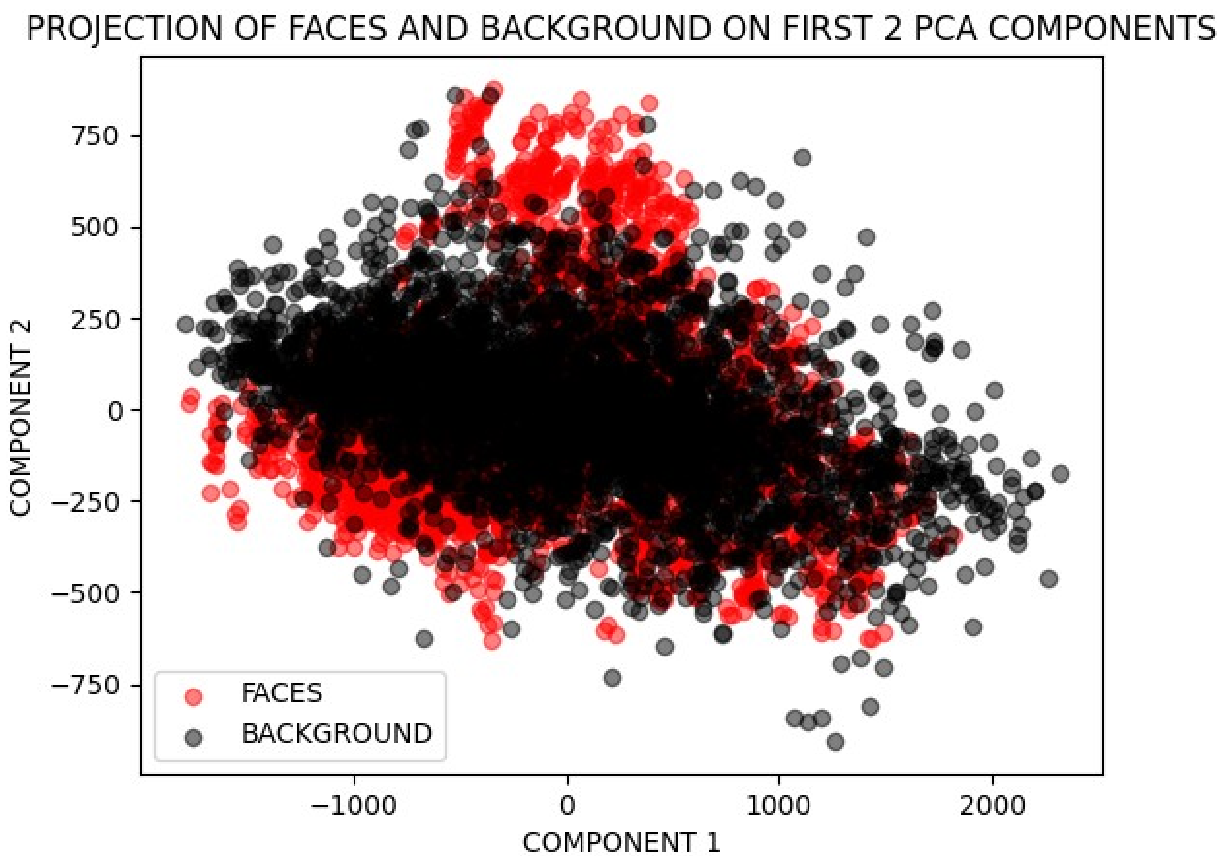
```
plt.ylabel('COMPONENT 2')
plt.title('FACES PROJECTED ONTO FIRST 2 PCA COMPONENTS')
plt.show()
```



c)

```
path = "C:/Users/rique/Downloads/datasets/backgrounds"

backgrounds = []

backgrounds = load_data(path, backgrounds)

background_matrix = get_matrix(backgrounds)

# APPLY PCA FROM FACE DATA
third_pca = PCA(n_components=2)
third_pca.fit(face_matrix)

projected_faces = third_pca.transform(face_matrix)
projected_background = third_pca.transform(background_matrix)

# PLOT
```

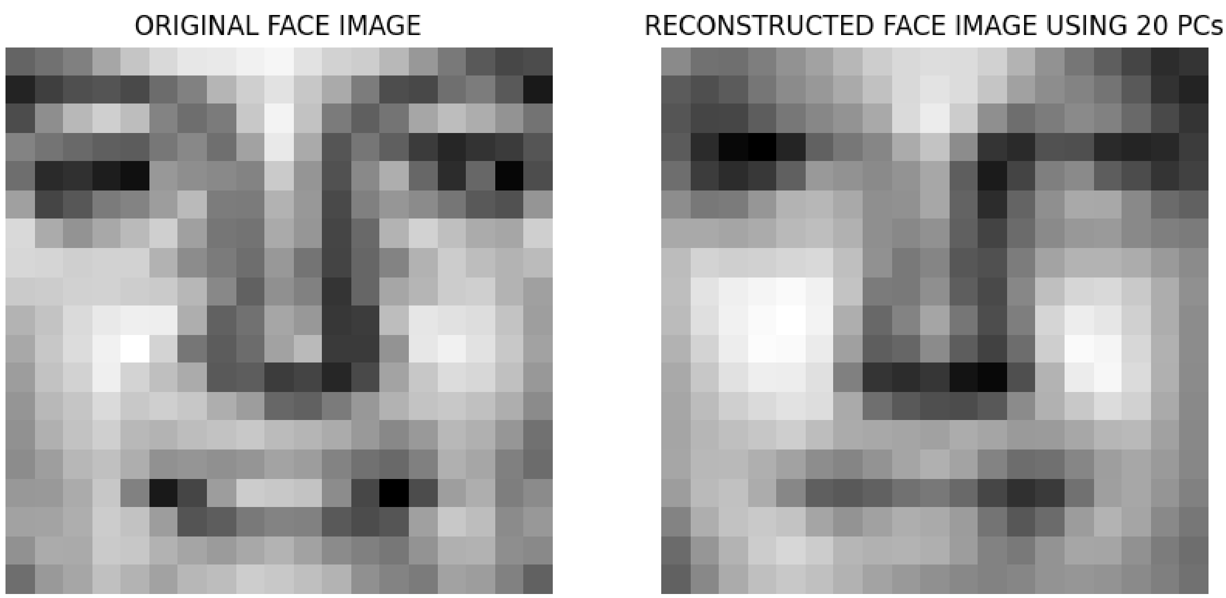
```
plt.scatter(projected_faces[:, 0], projected_faces[:, 1], alpha=0.5,
color='red', label='FACES')
plt.scatter(projected_background[:, 0], projected_background[:, 1],
alpha=0.5, color='black', label='BACKGROUND')
plt.xlabel('COMPONENT 1')
plt.ylabel('COMPONENT 2')
plt.title('PROJECTION OF FACES AND BACKGROUND ON FIRST 2 PCA
COMPONENTS')
plt.legend()
plt.show()
```



d)

```
# LOADING SPECIFIC FACE IMAGE
specific_face_path =
"C:/Users/rique/Downloads/datasets/faces/face00067.pgm"
face_img = Image.open(specific_face_path)

# FLATTEN IMAGE AND SUBTRACT MEAN
face_flat_img = np.array(face_img).flatten()
face_flat_img_centered = face_flat_img - np.mean(face_flat_img)
```

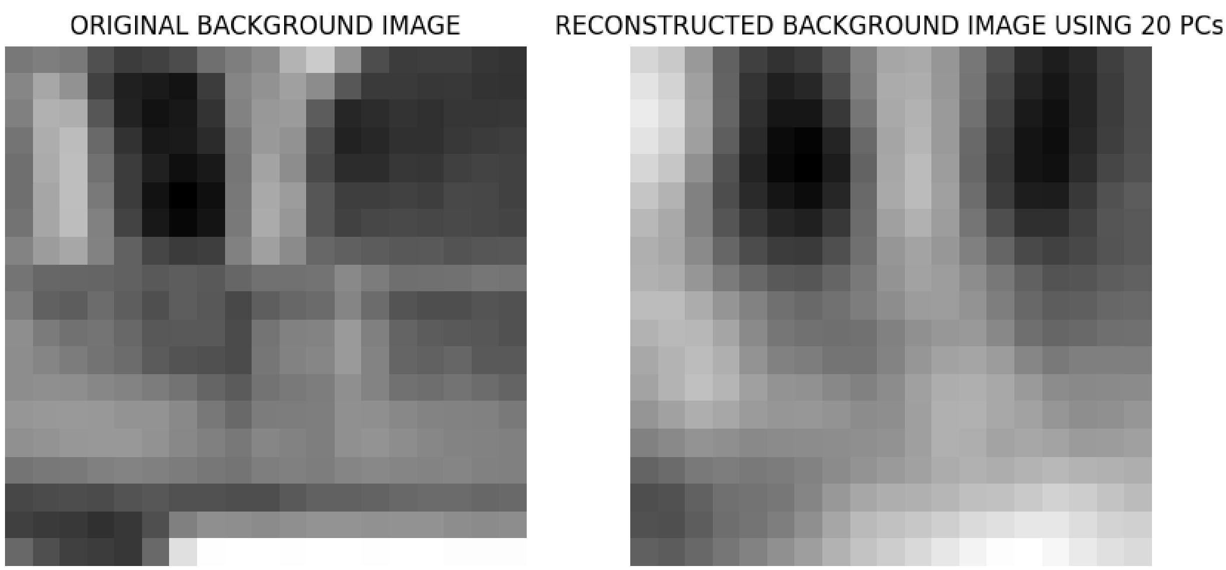
```
pca = PCA(n_components=20)
pca.fit(face_matrix)

transformed_face_img = pca.transform([face_flat_img_centered])
reconstructed_face_img = pca.inverse_transform(transformed_face_img)

# RESHAPE THE RECONSTRUCTRED FACE IMAGE TO ITS ORIGINAL SHAPE
reconstructed_face_img = reconstructed_face_img.reshape(face_img.size)

# PLOT
fig, axes = plt.subplots(1, 2, figsize=(10, 5))
axes[0].imshow(face_img, cmap='gray')
axes[0].set_title('ORIGINAL FACE IMAGE')
axes[0].axis('off')

axes[1].imshow(reconstructed_face_img, cmap='gray')
axes[1].set_title('RECONSTRUCTED FACE IMAGE USING 20 PCs')
axes[1].axis('off')

plt.show()
```



e)

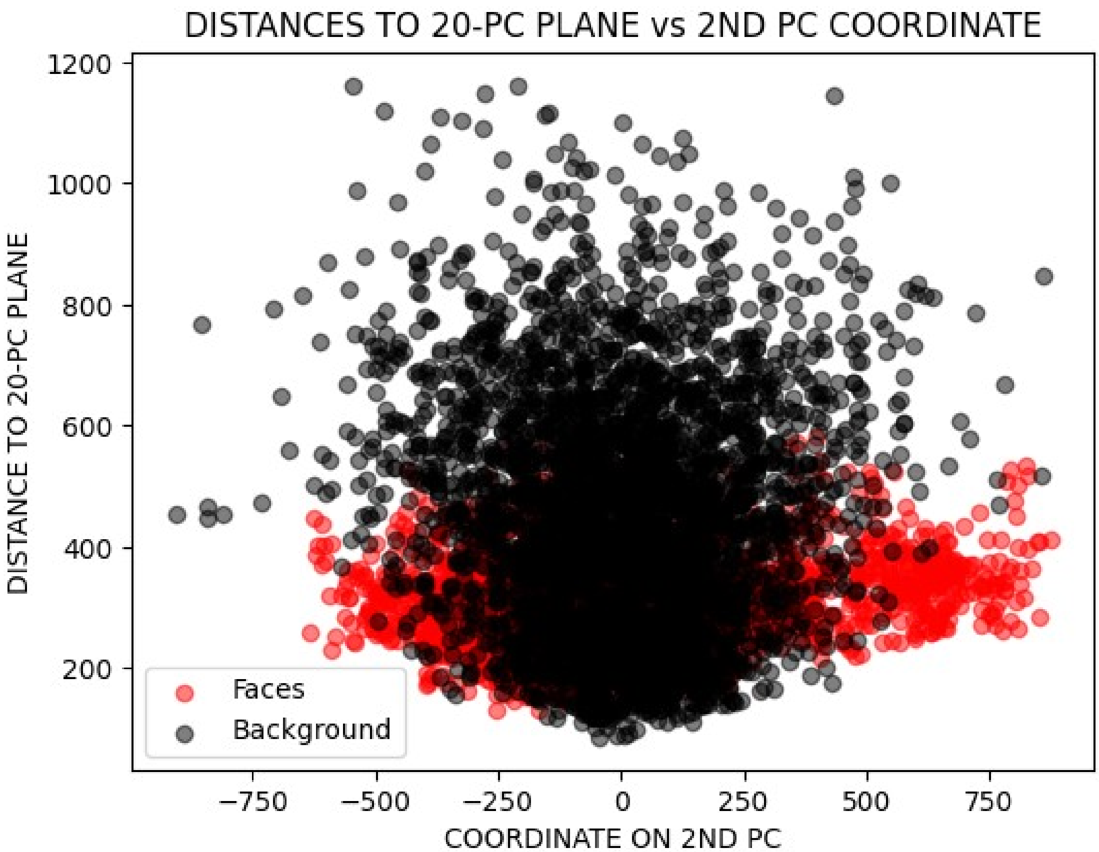
```
specific_background_path =
"C:/Users/rique/Downloads/datasets/backgrounds/B1_00192.pgm"
background_img = Image.open(specific_background_path)

# FLATTEN IMAGE AND SUBTRACT MEAN
```

```
bg_flat_img = np.array(background_img).flatten()
bg_flat_img_centered = bg_flat_img - np.mean(bg_flat_img)

# USING PCA WITH 20 COMPONENTS (PCA MODEL FROM PART A - 20 COMPONENTS)
pca = PCA(n_components=20)
pca.fit(background_matrix)

transformed_background_img = pca.transform([bg_flat_img_centered])
reconstructed_background_img =
pca.inverse_transform(transformed_background_img)

# RESHAPE THE RECONSTRUCTRED BACKGROUND IMAGE TO ITS ORIGINAL SHAPE
reconstructed_background_img =
reconstructed_background_img.reshape(background_img.size)

# PLOT
fig, axes = plt.subplots(1, 2, figsize=(10, 5))
axes[0].imshow(background_img, cmap='gray')
axes[0].set_title('ORIGINAL BACKGROUND IMAGE')
axes[0].axis('off')

axes[1].imshow(reconstructed_background_img, cmap='gray')
axes[1].set_title('RECONSTRUCTED BACKGROUND IMAGE USING 20 PCs')
axes[1].axis('off')

plt.show()
```



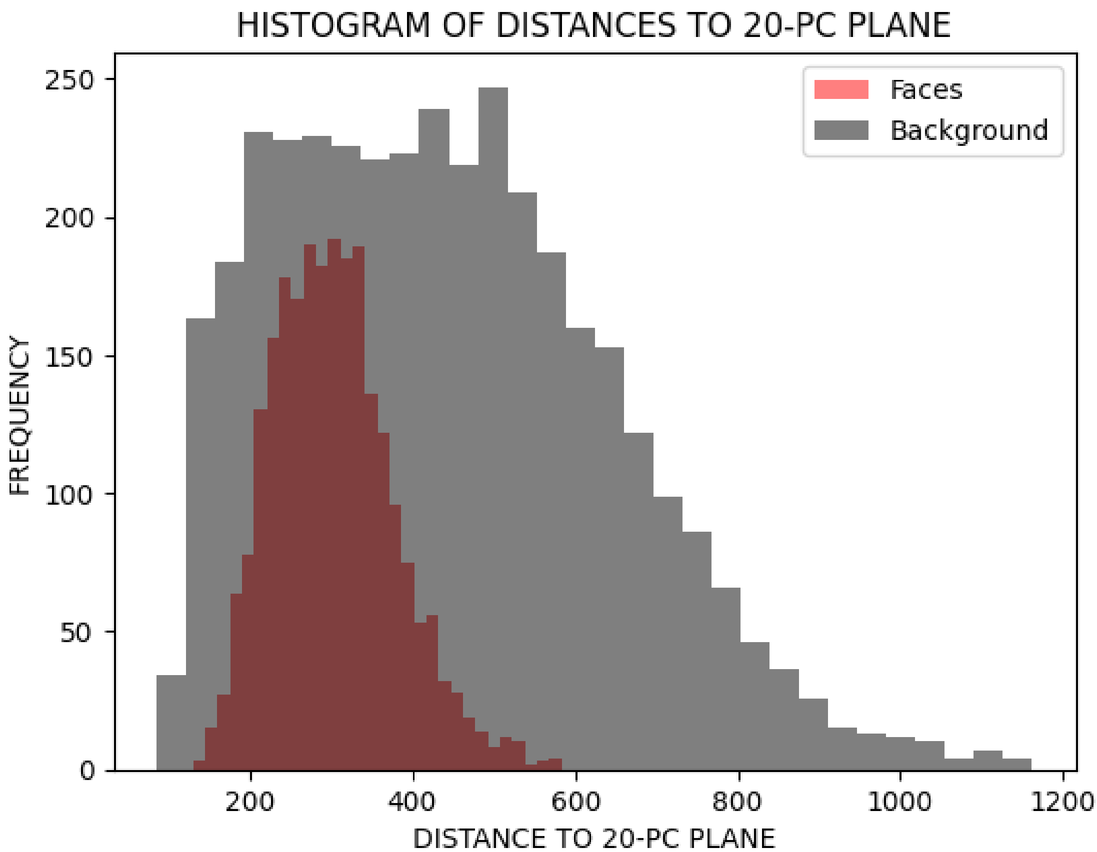
f)

```
# APPLYING PCA WITH 20 COMPONENTS
fourth_pca_20 = PCA(n_components=20)
fourth_pca_20.fit(face_matrix) # TRAINING ONLY ON FACE DATA

# PROJECT BOTH DATASETS ONTO THE 20 PCs
projected_faces = fourth_pca_20.transform(face_matrix)
projected_background = fourth_pca_20.transform(background_matrix)

# RECONSTRUCTING IMAGES FROM THE 20 PCs
reconstructed_faces = fourth_pca_20.inverse_transform(projected_faces)
reconstructed_background =
fourth_pca_20.inverse_transform(projected_background)

# CALCULATING DISTANCES TO THE 20-PC PLANE
distances_faces = np.sqrt(np.sum((face_matrix -
reconstructed_faces)**2, axis=1))
distances_background = np.sqrt(np.sum((background_matrix -
reconstructed_background)**2, axis=1))

# COORDINATES ON THE 2ND PC
second_pc_faces = projected_faces[:, 1]
second_pc_background = projected_background[:, 1]

# PLOT
plt.scatter(second_pc_faces, distances_faces, alpha=0.5, color='red',
label='Faces')
plt.scatter(second_pc_background, distances_background, alpha=0.5,
color='black', label='Background')
plt.xlabel('COORDINATE ON 2ND PC')
plt.ylabel('DISTANCE TO 20-PC PLANE')
plt.title('DISTANCES TO 20-PC PLANE vs 2ND PC COORDINATE')
plt.legend()
plt.show()
```



g)

```
# HISTOGRAM PLOT
plt.hist(distances_faces, bins=30, alpha=0.5, color='red',
label='Faces')
plt.hist(distances_background, bins=30, alpha=0.5, color='black',
label='Background')
plt.xlabel('DISTANCE TO 20-PC PLANE')
plt.ylabel('FREQUENCY')
plt.title('HISTOGRAM OF DISTANCES TO 20-PC PLANE')
plt.legend()
plt.show()
```


HISTOGRAM OF DISTANCES TO 20-PC PLANE
Faces
Background
FREQUENCY
DISTANCE TO 20-PC PLANE
0
50
100
150
200
250
200
400
600
800
1000
1200