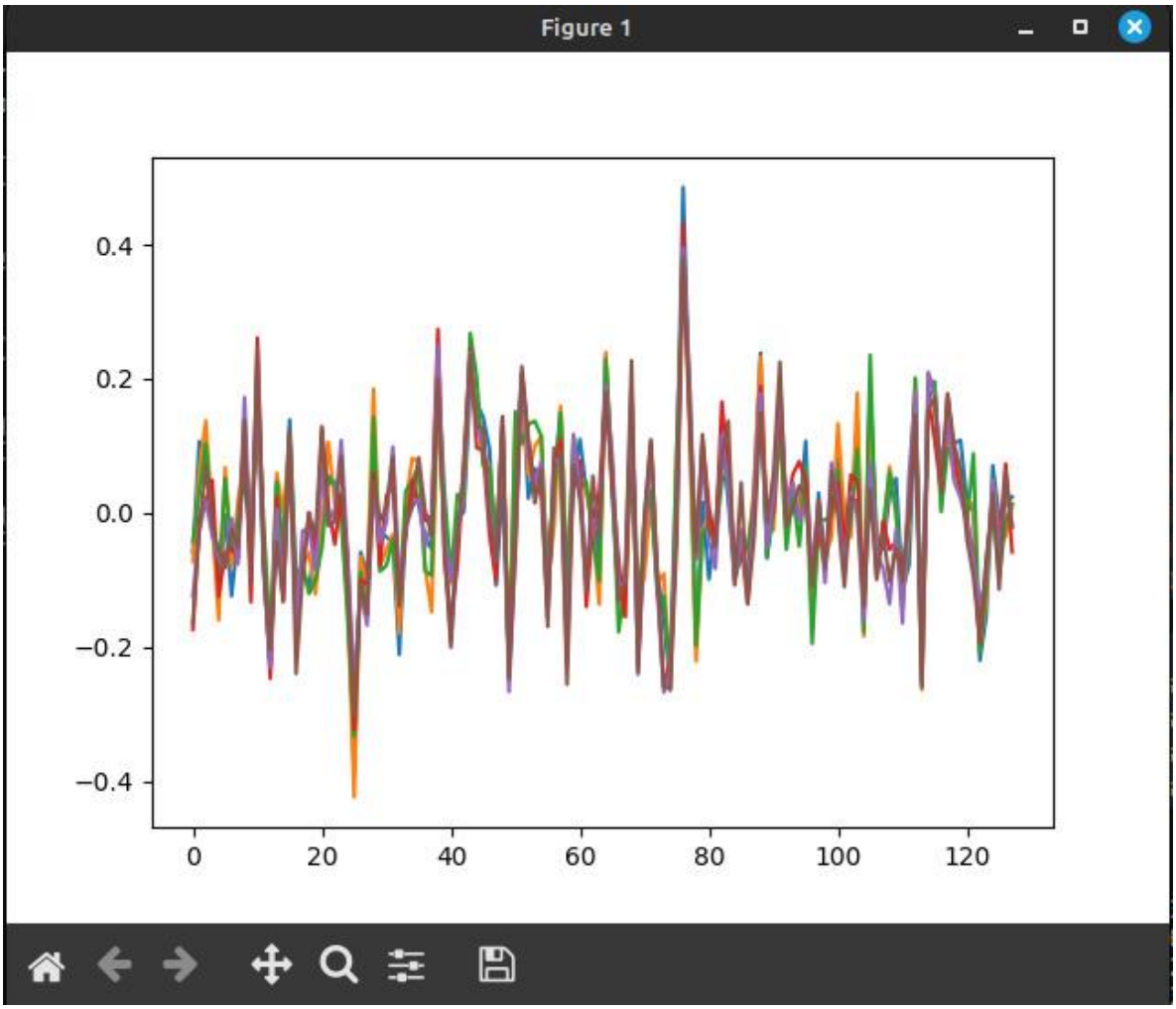
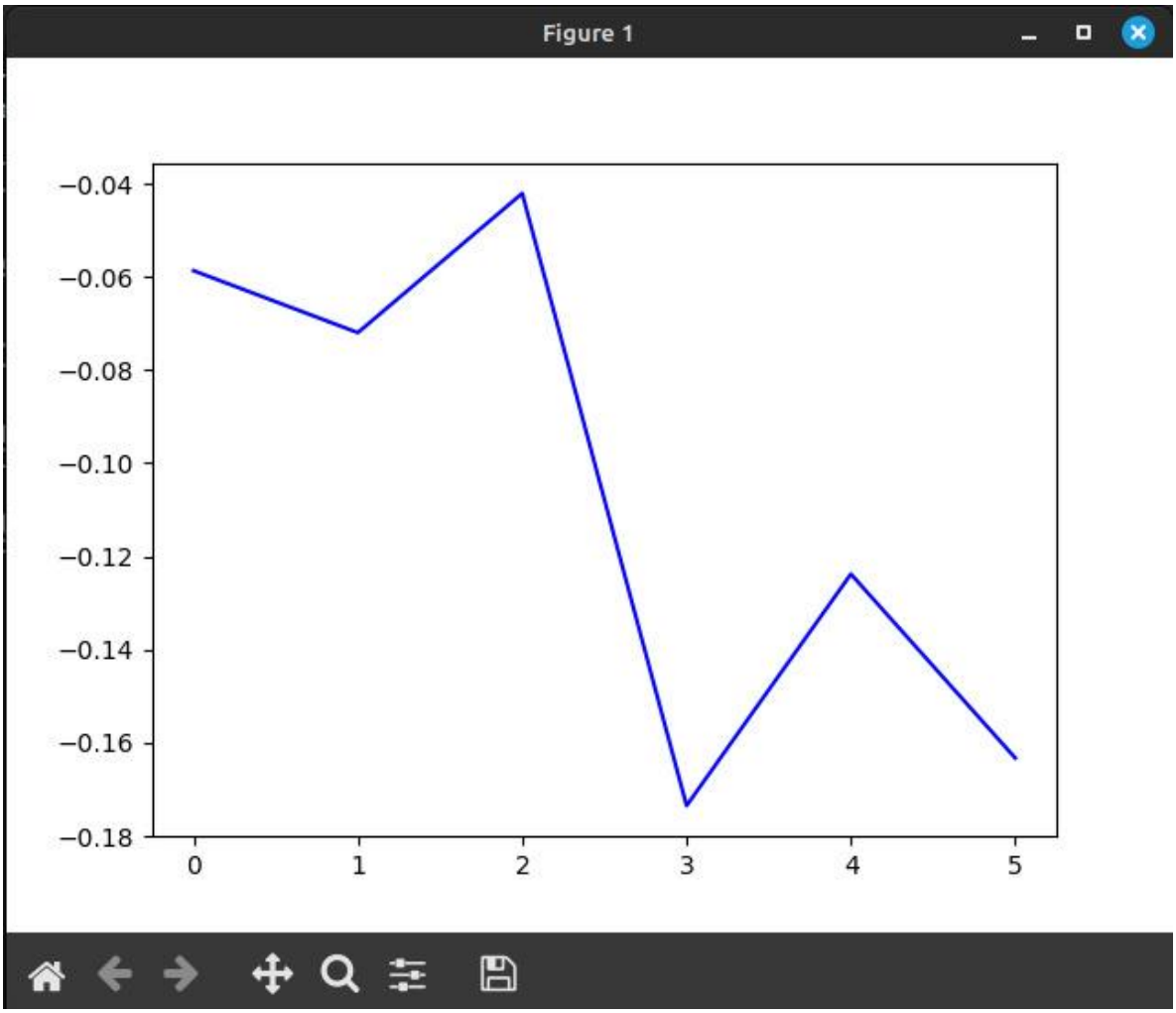
```
#Librerias
import face_recognition
import cv2
import numpy as np
from matplotlib import pyplot as plt

# Load image:
img5 = face_recognition.load_image_file("5.jpg")
img6 = face_recognition.load_image_file("6.jpg")
img7 = face_recognition.load_image_file("7.jpg")
img8 = face_recognition.load_image_file("8.jpg")
img9 = face_recognition.load_image_file("9.jpg")
img10 = face_recognition.load_image_file("10.jpg")

# Calculate the encodings for every face of the image:
encodings5 = face_recognition.face_encodings(img5)
encodings6 = face_recognition.face_encodings(img6)
encodings7 = face_recognition.face_encodings(img7)
encodings8 = face_recognition.face_encodings(img8)
encodings9 = face_recognition.face_encodings(img9)
encodings10 = face_recognition.face_encodings(img10)

# Show the first encoding:

print(len(encodings5))
print(encodings5[0])
print('')
print(len(encodings6))
print(encodings6[0])
print('')
print(len(encodings7))
print(encodings7[0])
print('')
print(len(encodings8))
print(encodings8[0])
print('')
print(len(encodings9))
print(encodings9[0])
print('')
print(len(encodings10))
print(encodings10[0])
print('')
plt.plot(encodings5[0])
plt.plot(encodings6[0])
plt.plot(encodings7[0])
plt.plot(encodings8[0])
plt.plot(encodings9[0])
plt.plot(encodings10[0])
plt.show()

# Create data (three different 'clusters' of points (it should be of
np.float32 data type):
```

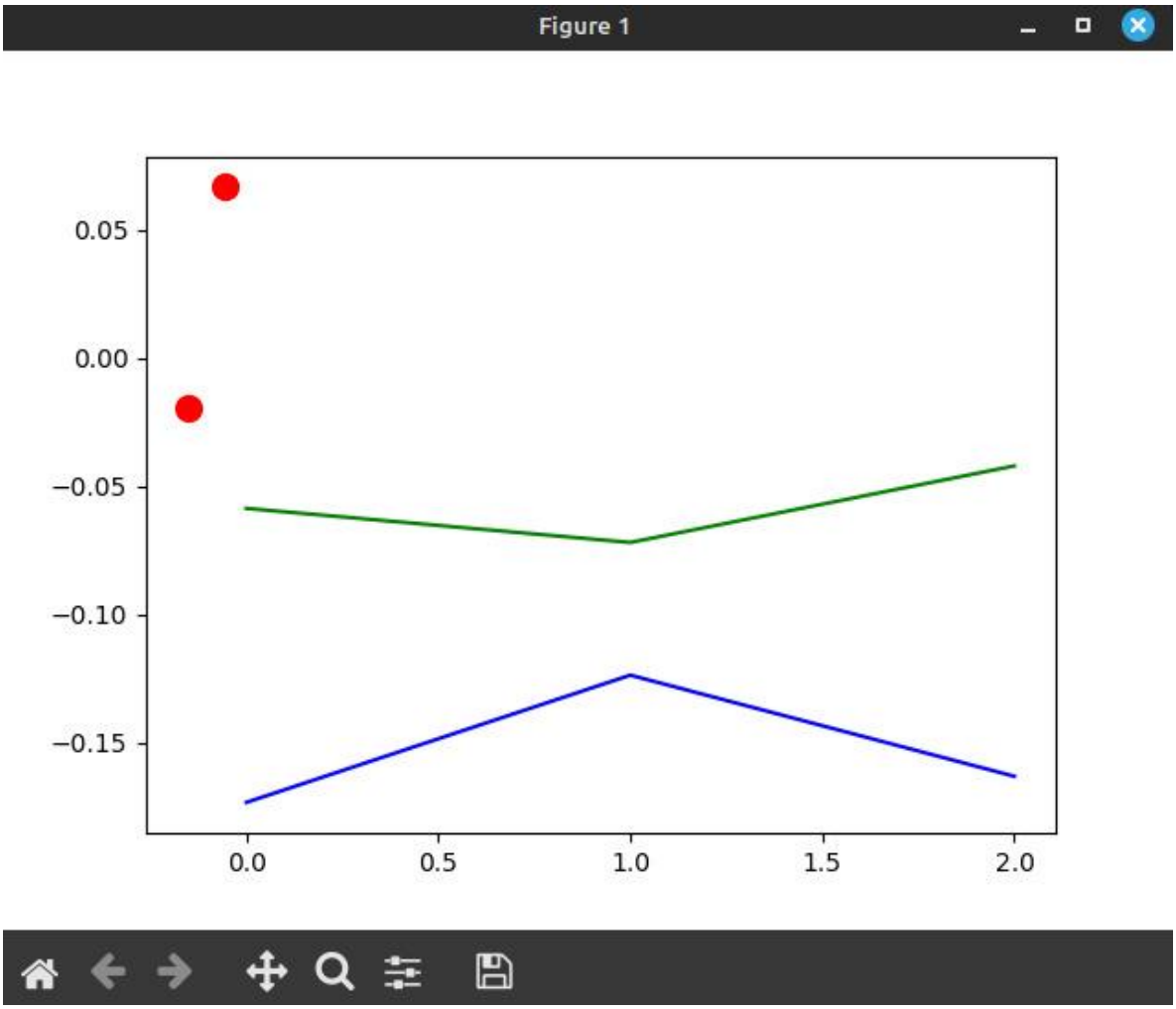
```
data = np.float32(np.vstack((
    (encodings5[0]),
    (encodings6[0]),
    (encodings7[0]),
    (encodings8[0]),
    (encodings9[0]),
    (encodings10[0]))))
plt.plot(data[:, 0], c='b')
plt.show()

# K means
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 20,
1.0)
ret, label, center = cv2.kmeans(data, 2, None, criteria, 10,
cv2.KMEANS_RANDOM_CENTERS)

# Now separate the data using label output
A = data[label.ravel() == 0]
B = data[label.ravel() == 1]

# plot it
plt.plot(A[:, 0],  c='b')
  plt.plot(B[:, 0],  c='g')
  plt.scatter(center[:, 0], center[:, 1], s=100, c='r')
  plt.show()
```



“Encodings” de las imágenes, una grafica que junta todo el “encodings” de todas las imágenes juntas



Vector resultante de los “encodings”



Algoritmo K-means

Este código utiliza las siguientes librerías: face_recognition, cv2 (OpenCV), numpy y matplotlib.pyplot. A continuación, se resumen las acciones realizadas por el código:

1. Se cargan seis imágenes (5.jpg, 6.jpg, 7.jpg, 8.jpg, 9.jpg y 10.jpg) utilizando la función face_recognition.load_image_file().
2. Se calculan las codificaciones de rostros para cada imagen utilizando la función face_recognition.face_encodings(). Estas codificaciones representan las características distintivas de cada rostro detectado en las imágenes.

3. Se imprimen las longitudes de las codificaciones y se muestra la primera codificación de cada imagen en la consola.
4. Se utiliza matplotlib.pyplot para trazar las primeras codificaciones en un gráfico.
5. Se crea un conjunto de datos combinando las primeras codificaciones de las seis imágenes utilizando la función np.vstack() de la librería numpy.
6. Se traza un gráfico con los valores de la primera columna de los datos.
7. Se utiliza el algoritmo de K-means de OpenCV (cv2.kmeans()) para agrupar los datos en dos clústeres utilizando como criterio de parada el número máximo de iteraciones y un umbral de precisión.
8. Se separan los datos en dos grupos, A y B, según las etiquetas asignadas por el algoritmo K-means.
9. Se trazan dos gráficos separados para los grupos A y B, y se muestran los centros de cada clúster en rojo.

En resumen, este código carga imágenes, calcula codificaciones de rostros utilizando la biblioteca face_recognition, visualiza y analiza las codificaciones utilizando gráficos y aplica el algoritmo de K-means de OpenCV para agrupar los datos según las características de los rostros detectados.