## Importando Bibliotecas

```
%matplotlib inline
from PIL import Image
import matplotlib.pyplot as plt
import pandas as pd
from google.colab import files
import numpy as np
from numpy import linalg
from sklearn.decomposition import TruncatedSVD
import math
```

## Lendo fonte de dados do kaggle

```
df = pd.read_csv('/content/sample_data/weight-height.csv')
```

```
df.head()
```

| | Gender | Height | Weight |
|---|---|---|---|
| 0 | Male | 73.847017 | 241.893563 |
| 1 | Male | 68.781904 | 162.310473 |
| 2 | Male | 74.110105 | 212.740856 |
| 3 | Male | 71.730978 | 220.042470 |
| 4 | Male | 69.881796 | 206.349801 |

Next steps: Generate code with df | View recommended plots

```
df_male = df.loc[df['Gender']=='Male']
df_female = df.loc[df['Gender']=='Female']
```

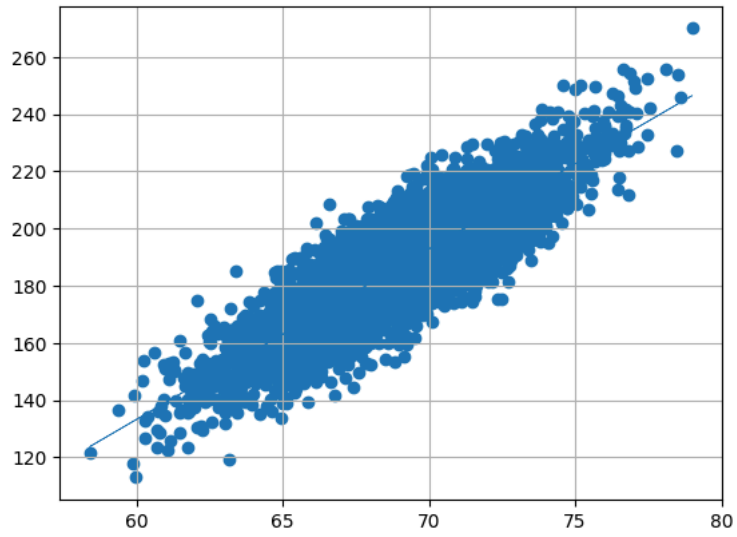
Construindo Matrizes A para os dados de cada gênero

```
A_matriz_male = []
A_matriz_female = []

for i in range(len(df_male)):
  A_matriz_male.append([df_male.iloc[i]['Height'], df_male.iloc[i]['Weight']])

A_male = np.array(A_matriz_male)

for i in range(len(df_female)):
  A_matriz_female.append([df_female.iloc[i]['Height'], df_female.iloc[i]['Weight']])

A_female = np.array(A_matriz_female)


print(A_male)
```

```
[[ 73.84701702 241.89356318]
 [ 68.78190405 162.31047252]
 [ 74.11010539 212.74085556]
 ...
 [ 67.01379497 199.19540008]
 [ 71.55771849 185.90590949]
 [ 70.35187988 198.90301194]]
```

```
U_male, S_male, VT_male = linalg.svd(A_male)
U_female, S_female, VT_female = linalg.svd(A_female)
```

```
x = A_male[:,0]
y = A_male[:,1]
m, b = np.polyfit(x,y, 1)

fig = plt.figure()
ax = fig.gca()
# ax.set_xticks(np.arange(0, 50, 100))
# ax.set_yticks(np.arange(50, 150, 300))
plt.scatter(x, y, marker="o")

plt.plot(x, m*x + b,linewidth=0.5)
# plt.xlim(0,300)
# plt.ylim(50,300)
plt.grid()
plt.show()
```




## Valores Singulares da Matriz com os dados relacionados ao gênero Masculino

```
for i in S_male:
  print(i)
```

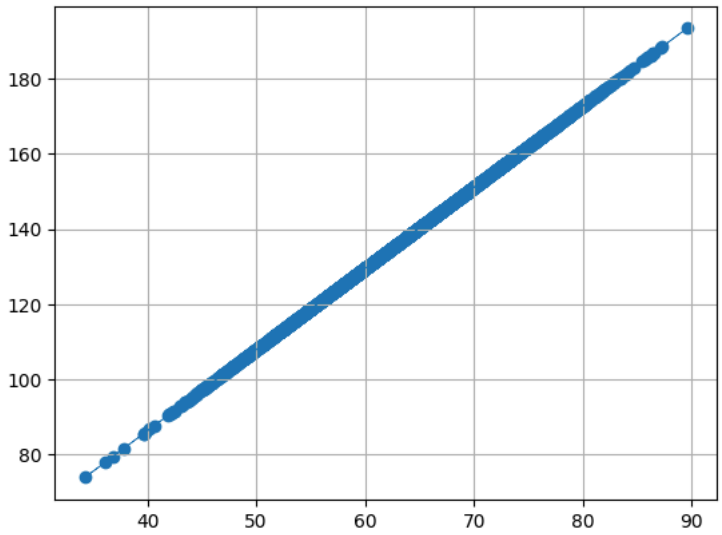
```
    14163.082890099258
    333.00607247678
```

## Matriz Aproximada

```
u1 = np.array(U_male[:,0])
u2 = np.array(U_male[:,1])
v1T = np.array(VT_male[0,:])
v2T = np.array(VT_male[1,:])
sigma1 = S_male[0]
sigma2 = S_male[1]
A1 = sigma1*np.outer(u1,v1T)
print(A1)
```

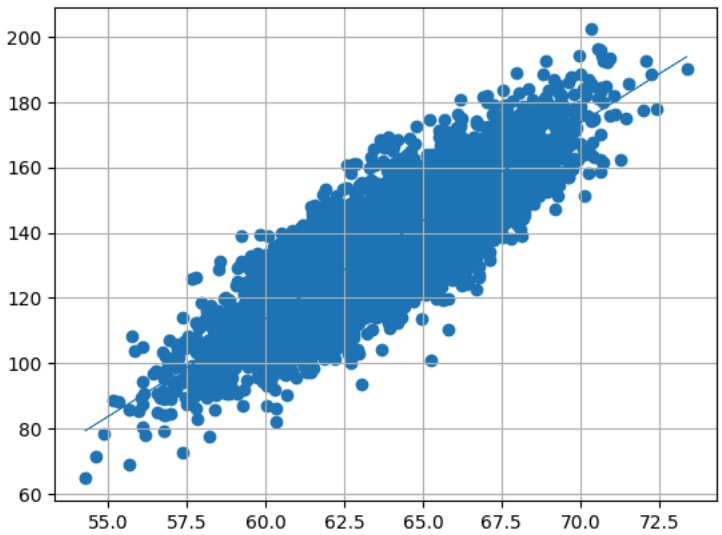
```
    [[ 86.92379558 237.09944299]
     [ 60.60428386 165.30849637]
     [ 77.53349346 211.48579615]
     ...
     [ 72.31514247 197.25185588]
     [ 68.55865783 187.00540484]
     [ 72.61614851 198.07290108]]
```

```
x = A1[:,0]
y = A1[:,1]
m, b = np.polyfit(x,y, 1)

fig = plt.figure()
ax = fig.gca()
# ax.set_xticks(np.arange(0, 50, 100))
# ax.set_yticks(np.arange(50, 150, 300))
plt.scatter(x, y, marker="o")

plt.plot(x, m*x + b,linewidth=0.5)
# plt.xlim(0,300)
# plt.ylim(50,300)
plt.grid()
plt.show()
```




## Valores Singulares da Matriz com os dados relacionados ao gênero Feminino

```
x = A_female[:,0]
y = A_female[:,1]
m, b = np.polyfit(x,y, 1)

fig = plt.figure()
ax = fig.gca()
# ax.set_xticks(np.arange(0, 50, 100))
# ax.set_yticks(np.arange(50, 150, 300))
plt.scatter(x, y, marker="o")

plt.plot(x, m*x + b,linewidth=0.5)
# plt.xlim(0,300)
# plt.ylim(50,300)
plt.grid()
plt.show()
```



```
for i in S_female:
  print(i)
```

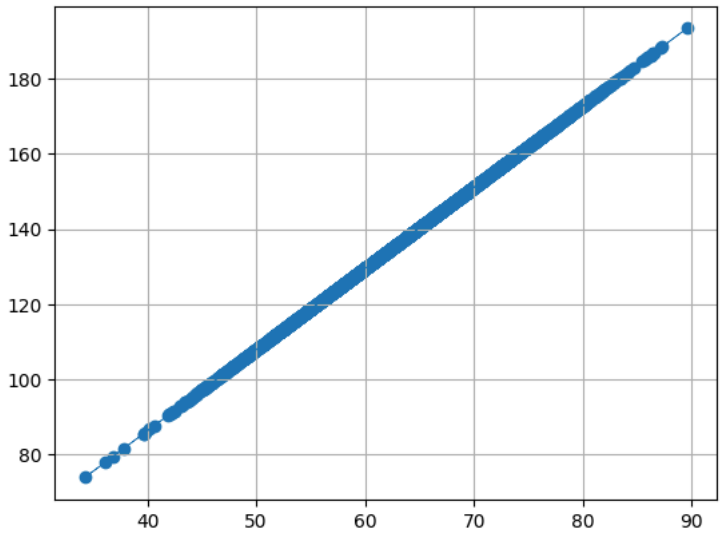
```
    10688.466047930735
    431.05578843749976
```

## Matriz Aproximada

```
u1 = np.array(U_female[:,0])
u2 = np.array(U_female[:,1])
v1T = np.array(VT_female[0,:])
v2T = np.array(VT_female[1,:])
sigma1 = S_female[0]
sigma2 = S_female[1]
A1 = sigma1*np.outer(u1,v1T)
print(A1)
```

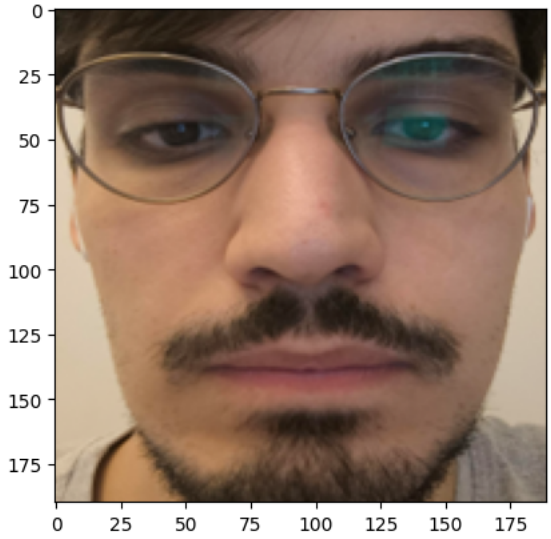
```
    [[ 49.33553423 106.52302487]
     [ 65.40702079 141.22384222]
     [ 61.16405015 132.06261444]
     ...
     [ 60.27361749 130.14003303]
     [ 74.67695956 161.23906926]
     [ 54.27994884 117.19877834]]
```

```
x = A1[:,0]
y = A1[:,1]
m, b = np.polyfit(x,y, 1)

fig = plt.figure()
ax = fig.gca()
# ax.set_xticks(np.arange(0, 50, 100))
# ax.set_yticks(np.arange(50, 150, 300))
plt.scatter(x, y, marker="o")

plt.plot(x, m*x + b,linewidth=0.5)
# plt.xlim(0,300)
# plt.ylim(50,300)
plt.grid()
plt.show()
```



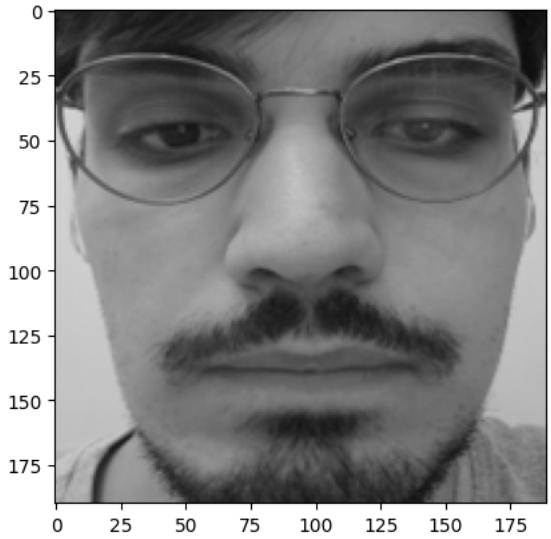

## Lendo imagem

```
img = Image.open('/content/sample_data/imagem_rosto.png')
plt.imshow(img);
```

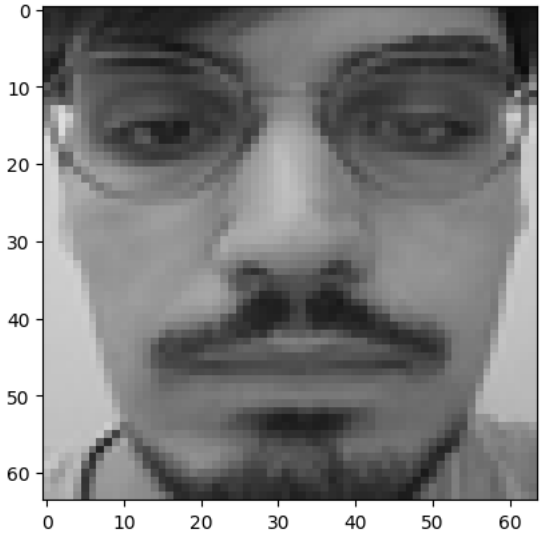



```
imggray = img.convert('LA')
print(np.shape(imggray))
print(imggray)

plt.imshow(imggray, cmap='gray');
```

```
(190, 189, 2)
<PIL.Image.Image image mode=LA size=189x190 at 0x79FF6D81B040>
```




```
img_64 = imggray.resize((64, 64), Image.Resampling.LANCZOS)
plt.imshow(img_64);
```

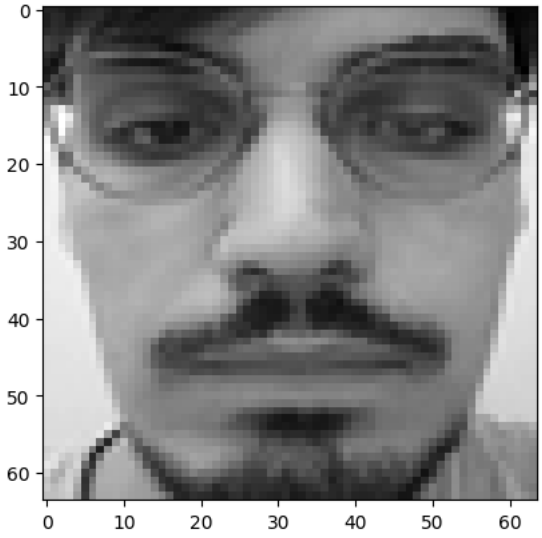


```
img_64 = np.array(list(img_64.getdata(band=0)), float)
img_64.shape = (64, 64)
```

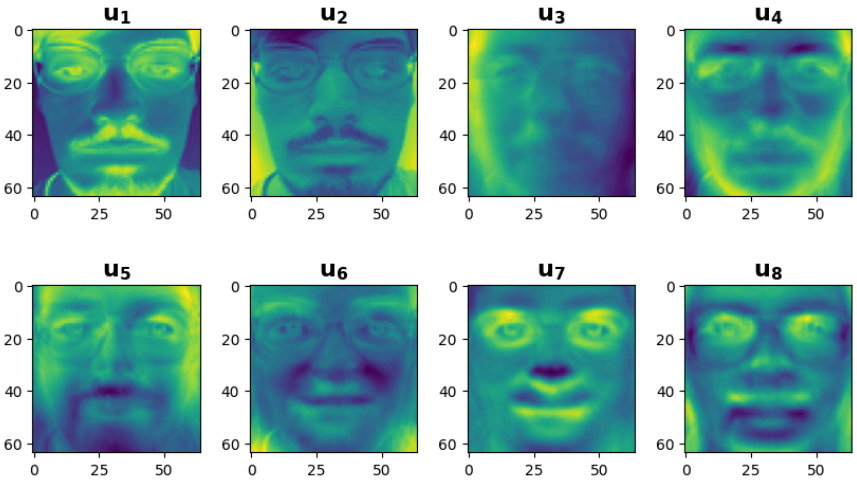
## Incluindo minha imagem no vetor de eigenfaces

```
from sklearn.datasets import fetch_olivetti_faces
data = fetch_olivetti_faces()
imgs = data.images
imgs = np.append(imgs, [img_64], axis=0)
nimg,width,height=imgs.shape

M = imgs.reshape((-1, width*height)).T
x= np.zeros((nimg, 1))

# escolha a face
facenumber = 400
x[facenumber, 0] = 1

y = M @ x
print(M.shape,x.shape,y.shape)
plt.imshow(y.reshape((width,height)), cmap='gray')
plt.show()
```

```
(4096, 401) (401, 1) (4096, 1)
```

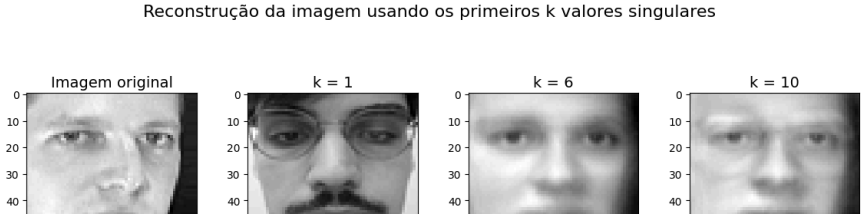



```
U, s, VT = linalg.svd(M)

fig, axes = plt.subplots(2, 4, figsize=(10,6))
plt.subplots_adjust(wspace=0.3, hspace=0.1)

for i in range(0, 8):
 axes[i // 4, i % 4].imshow(U[:, i].reshape((width,height)))
 axes[i // 4, i % 4].set_title("$\mathbf{{u_{0}}}$".format(i+1), fontsize=16)

plt.show()
```




Clique duas vezes (ou pressione "Enter") para editar

```
x= np.zeros((nimg, 1))
facenumber = 1
x[facenumber, 0] = 1
Sigma = np.zeros((M.shape[0], M.shape[1]))
Sigma[:min(M.shape[0], M.shape[1]), :min(M.shape[0], M.shape[1])] = np.diag(s)

fig, axes = plt.subplots(2, 4, figsize=(14, 8))
fig.suptitle("Reconstrução da imagem usando os primeiros k valores singulares", fontsize=16)
plt.subplots_adjust(wspace=0.3, hspace=0.1)

axes[0, 0].imshow(imgs[facenumber], cmap='gray')
axes[0, 0].set_title("Imagem original", fontsize=14)

k_list = [1, 6, 10, 15, 35, 50, 200]
for i in range(1, 8):
 # Reconstrução da matriz usando os primeiros k valores singulares
 k = k_list[i-1]
 mat_approx = U[:, :k] @ Sigma[:k, :k] @ VT[:k, :] @ x

 axes[i // 4, i % 4].imshow(mat_approx.reshape((width,height)), cmap='gray')
 axes[i // 4, i % 4].set_title("k = {}".format(k), fontsize=14)

plt.show()
```




```
x= np.zeros((nimg, 1))
facenumber = 400
x[facenumber, 0] = 1
Sigma = np.zeros((M.shape[0], M.shape[1]))
Sigma[:min(M.shape[0], M.shape[1]), :min(M.shape[0], M.shape[1])] = np.diag(s)

fig, axes = plt.subplots(2, 4, figsize=(14, 8))
fig.suptitle("Reconstrução da imagem usando os primeiros k valores singulares", fontsize=16)
plt.subplots_adjust(wspace=0.3, hspace=0.1)

axes[0, 0].imshow(imgs[facenumber], cmap='gray')
axes[0, 0].set_title("Imagem original", fontsize=14)

k_list = [1, 6, 10, 15, 35, 50, 200]
for i in range(1, 8):
 # Reconstrução da matriz usando os primeiros k valores singulares
 k = k_list[i-1]
 mat_approx = U[:, :k] @ Sigma[:k, :k] @ VT[:k, :] @ x

 axes[i // 4, i % 4].imshow(mat_approx.reshape((width,height)), cmap='gray')
 axes[i // 4, i % 4].set_title("k = {}".format(k), fontsize=14)

plt.show()
```