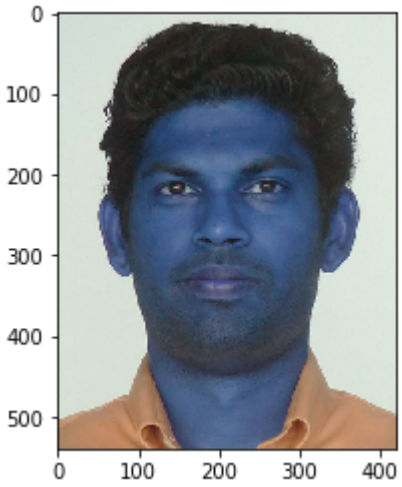
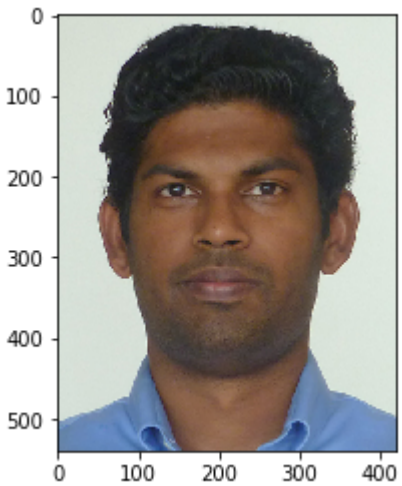
In [2]:
```
import numpy as np
import cv2
import matplotlib.pyplot as plt
import os
from sklearn.metrics import confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
import seaborn as sns
from sklearn.decomposition import PCA
import pandas as pd
```

In [33]:
```
#test images
image = cv2.imread('G:/Yamuna docs/College docs/Info Ret/Extra credit/photo1.jpg')
plt.imshow(image)
plt.show()
```

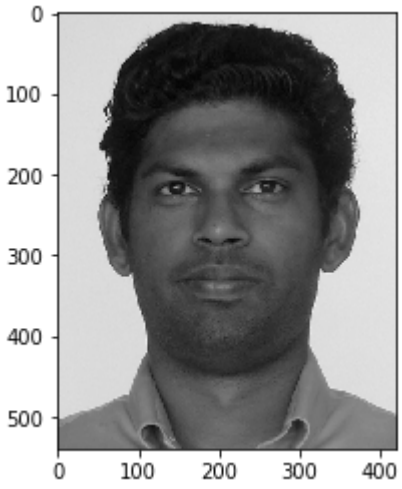


In [34]:
```
img_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
plt.imshow(img_rgb)
plt.show()
```

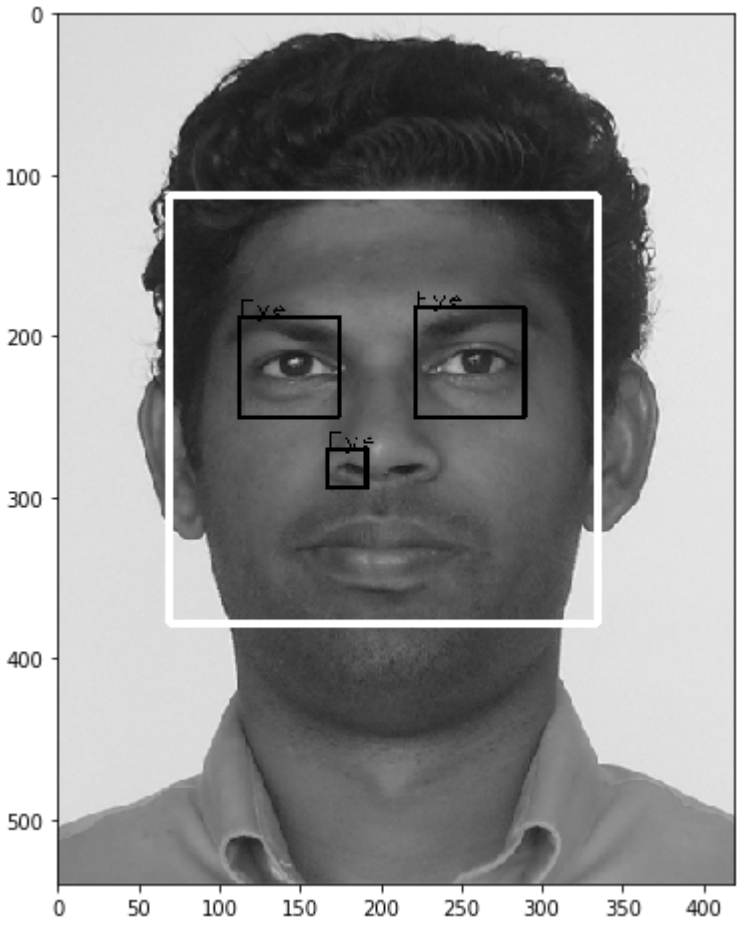


In [4]:
```
import numpy as np
import cv2

face_cascade = cv2.CascadeClassifier('C:/opencv/sources/data/haarcascades/haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('C:/opencv/sources/data/haarcascades/haarcascade_eye.xml')
smile_cascade = cv2.CascadeClassifier('C:/opencv/sources/data/haarcascades/haarcascade_smile.xml')
img = cv2.imread('G:/Yamuna docs/College docs/Info Ret/Extra credit/photo1.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
print(gray)
plt.imshow(gray,cmap='gray')
plt.show()
```

```
[[219 219 217 ... 229 229 230]
 [216 217 217 ... 228 227 228]
 [217 218 217 ... 226 228 229]
 ...
 [135 130 129 ... 137 135 141]
 [129 129 130 ... 137 136 134]
 [142 136 135 ... 142 143 149]]
```

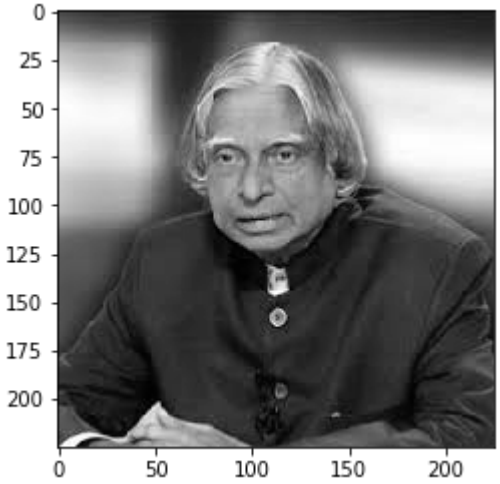


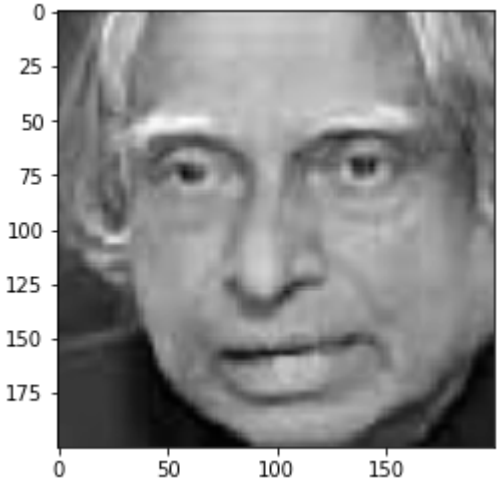
In [36]:
```
faces = face_cascade.detectMultiScale(gray,scaleFactor=1.1,minNeighbors=5,flag
s=cv2.CASCADE_SCALE_IMAGE)
for (x, y, w, h) in faces:
        if w > 250 :
            cv2.rectangle(gray, (x, y), (x+w, y+h), (255, 0, 0), 3)
            roi_gray = gray[y:y+h, x:x+w]
            roi_color = gray[y:y+h, x:x+w]
eyes = eye_cascade.detectMultiScale(roi_gray)
for (ex,ey,ew,eh) in eyes:
    cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)
    cv2.putText(gray,'Eye',(x + ex,y + ey), 1, 1, (0, 255, 0), 1)
plt.figure(figsize=(12,8))
plt.imshow(gray, cmap='gray')
plt.show()
```



In [19]:
```
str="G:\Yamuna docs\College docs\Info Ret\Extra credit\img/" #Image database f
or recognization

ncompnts=20
entries=os.listdir(str)
set=1
for entry in entries:
    if entry.endswith(".jpg") or entry.endswith(".jfif"):
#         print(entry)
        image = cv2.imread(str + entry)
#         plt.imshow(image)
#         plt.show()

        img_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
#         plt.imshow(img_rgb)
#         plt.show()

        img_gray=cv2.cvtColor(img_rgb,cv2.COLOR_BGR2GRAY)
        plt.imshow(img_gray,cmap=plt.get_cmap('gray'))
        plt.show()


        faces = face_cascade.detectMultiScale(img_gray, 1.3, 5)

        if len(faces)>0:

#         print(faces)
            for (x,y,w,h) in faces:
    #img_rgb = cv2.rectangle(img_gray,(x,y),(x+w,y+h),(255,0,0),2)
                roi_gray = img_gray[y:y+h, x:x+w]
#     roi_color = img_rgb[y:y+h, x:x+w]
    #eyes = eye_cascade.detectMultiScale(roi_gray)
#     for (ex,ey,ew,eh) in eyes:
#         cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)


            roi_gray_resize=cv2.resize(roi_gray,(200,200))
            plt.imshow(roi_gray_resize,cmap=plt.get_cmap('gray'))
            plt.show()

#         cv2.imshow('img',roi_gray_resize)
#         cv2.waitKey(0)
#         cv2.destroyAllWindows()

            fa=np.array(roi_gray_resize)
            fa=fa.flatten()
            if(set==1):
                dm=np.array(fa.T)
                m=entry.find('.')
                fname=entry[0:m-1]
                img_class=''.join([i for i in fname if not i.isdigit()])
                set=0;
            else:
                dm=np.column_stack((dm,fa.T))
                m=entry.find('.')
                s=entry[0:m]
```

```
                fname=np.column_stack((fname,s))
                s1=''.join([i for i in s if not i.isdigit()])
                img_class=np.column_stack((img_class,s1))

        else:
            print('Image not good',entry)
```





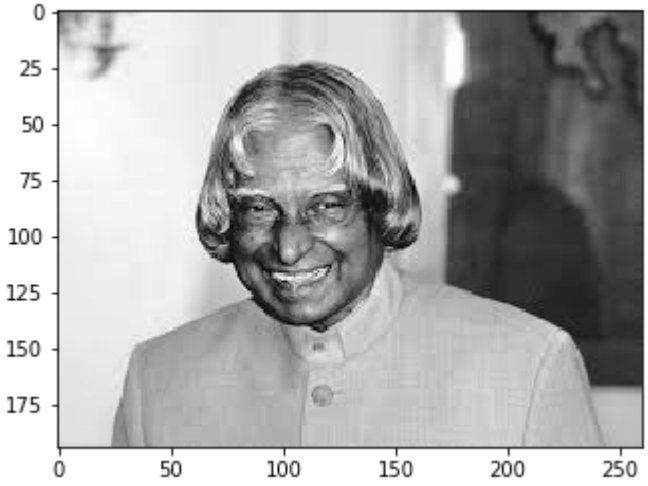


Image not good apj10.jpg

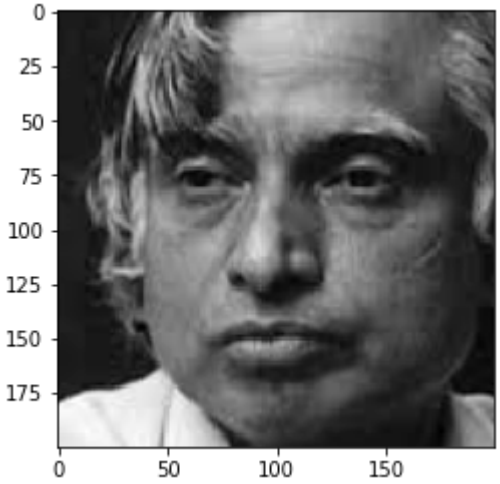
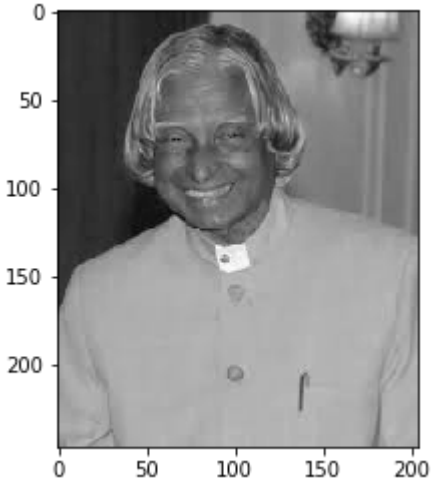
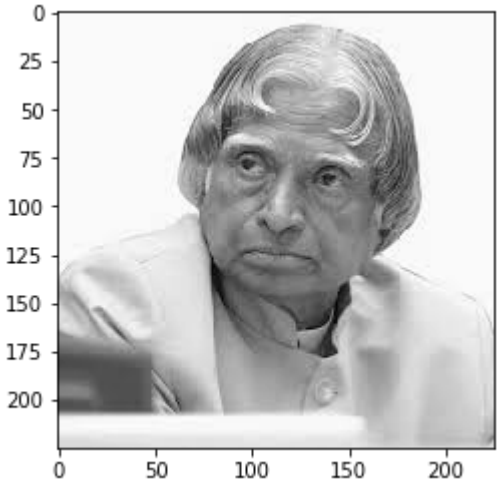


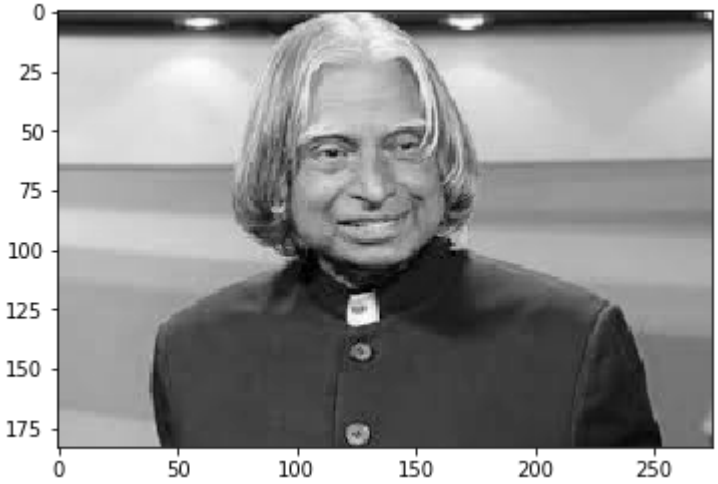
Image not good apj12.jpg




```
Image not good apj2.jfif
```




```
Image not good apj3.jfif
```



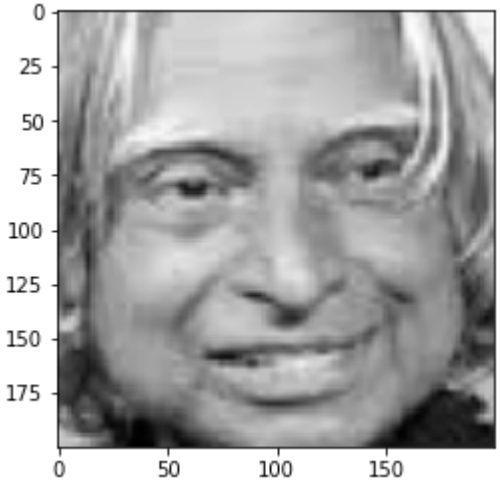
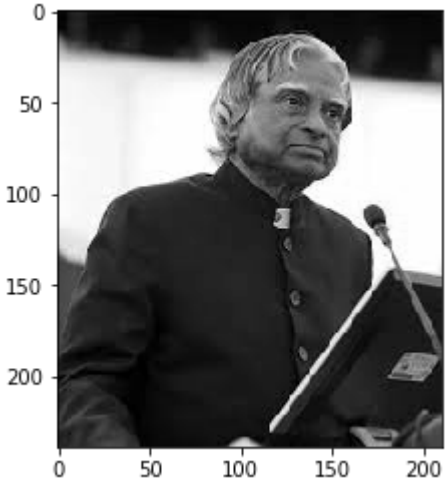
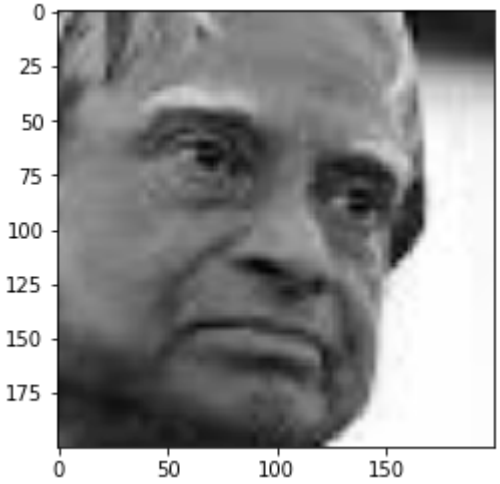


Image not good apj7.jpg





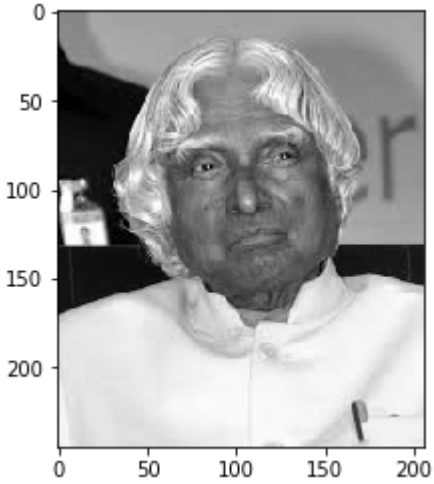
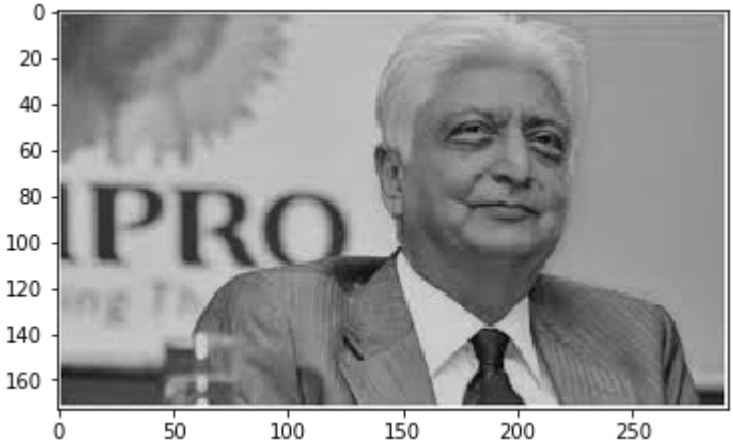
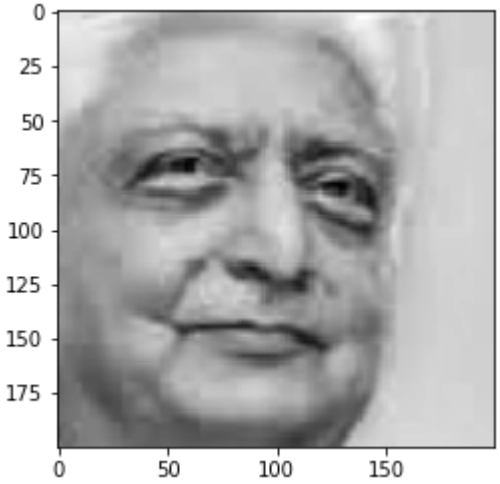


Image not good apj9.jpg





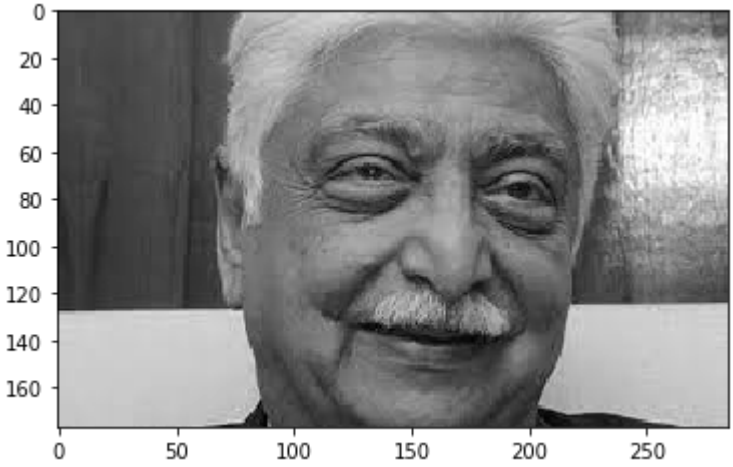
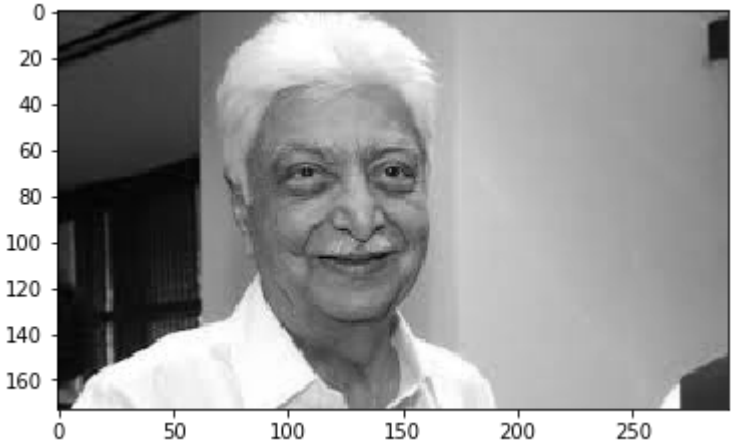
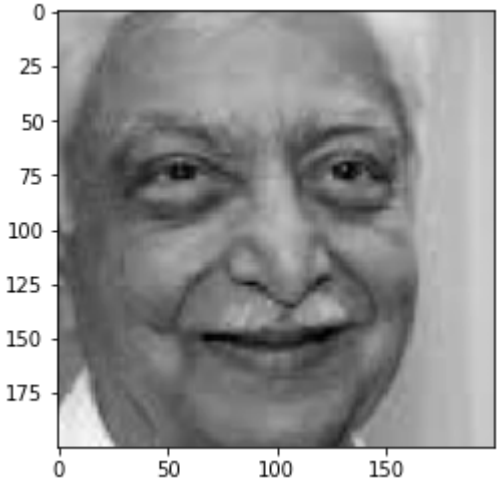
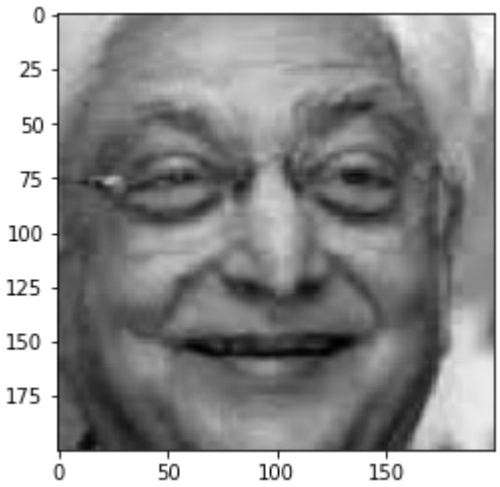


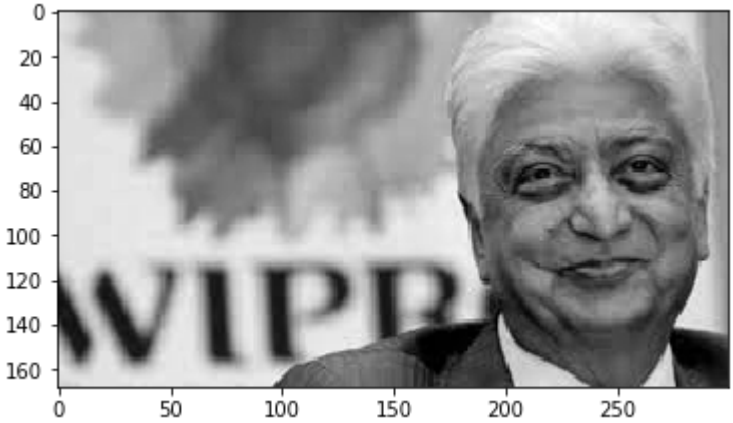
Image not good az10.jpg

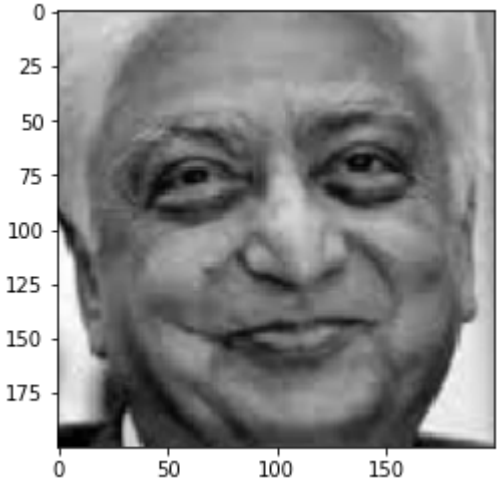



0
25
50
75
100
125
150
175
0
50
100
150

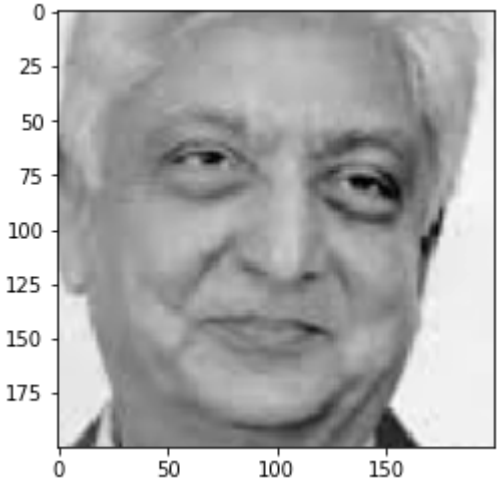

0
25
50
75
100
125
150
175
0
50
100
150
200
250
VIPRO


0
25
50
75
100
125
150
175
0
50
100
150

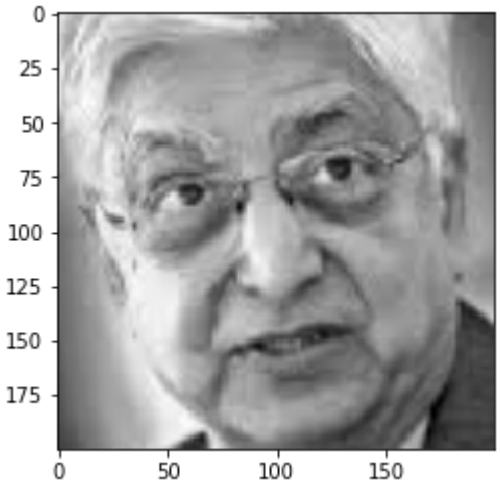

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150
200
250

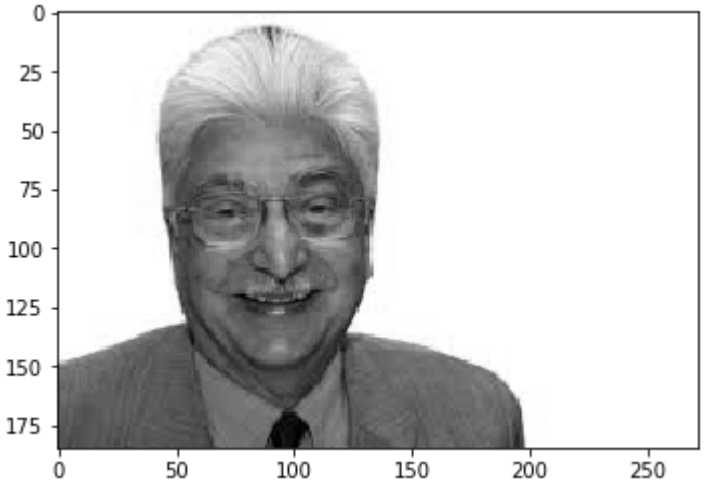

0
25
50
75
100
125
150
175
0
50
100
150

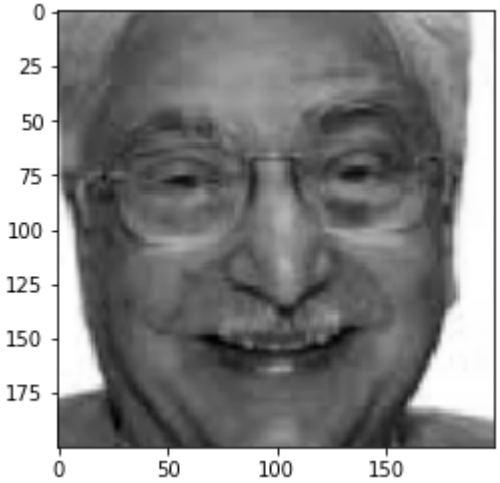

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250

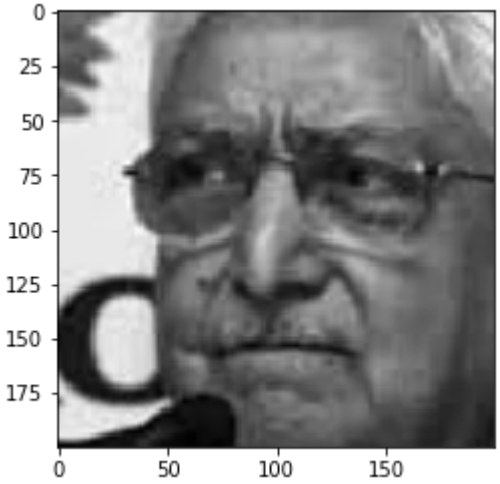

0
25
50
75
100
125
150
175
0
50
100
150


0
20
40
60
80
100
120
140
0
50
100
150
200
250
300

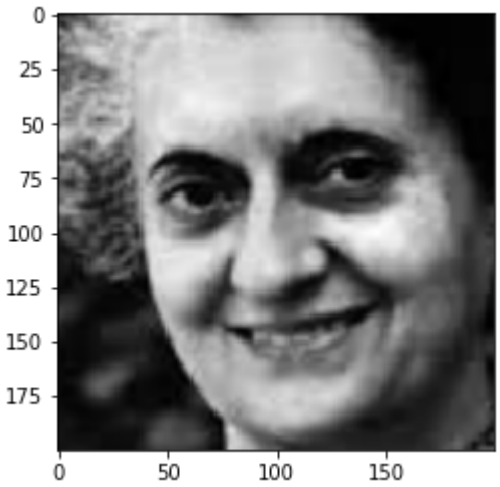
```
Image not good az6.jpg
```



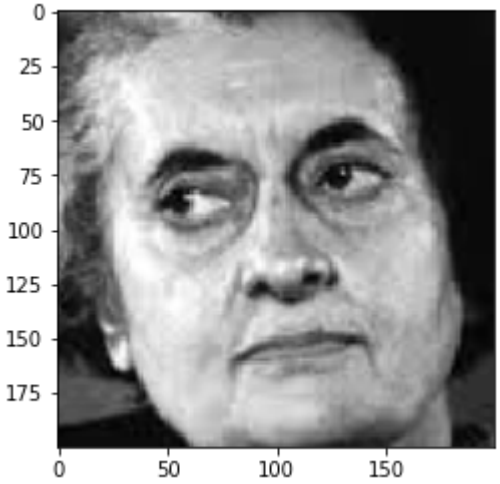


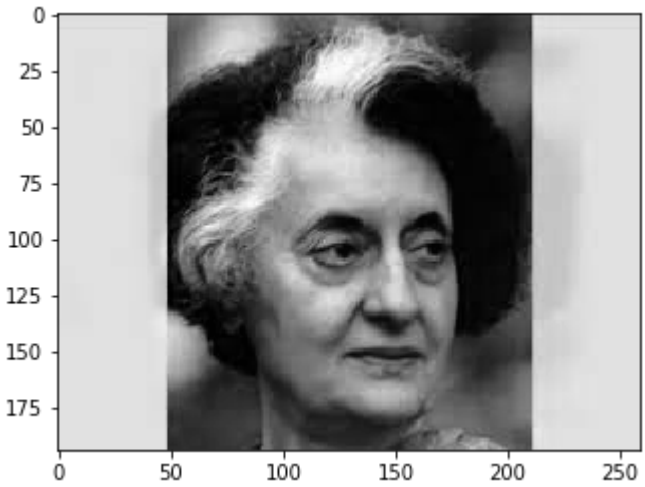

0
25
50
75
100
125
150
175
0
50
100
150

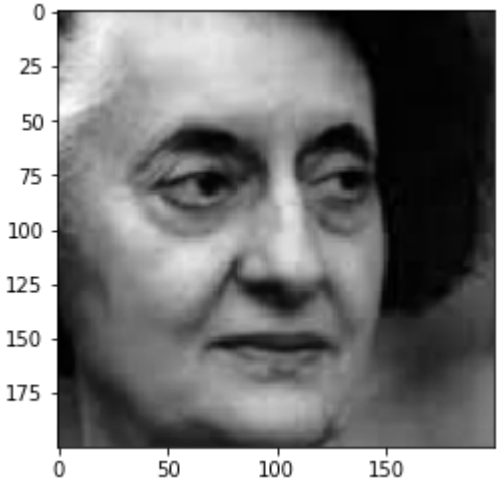

0
50
100
150
200
250
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

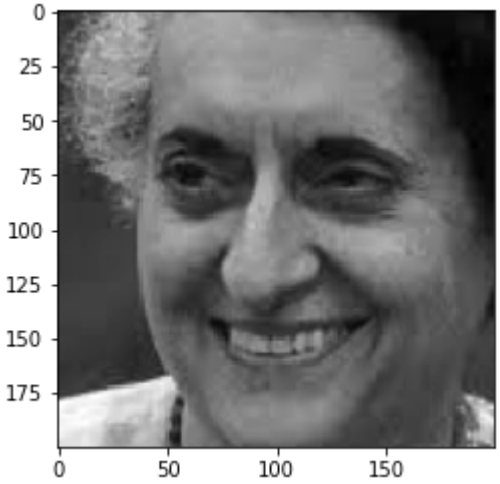

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

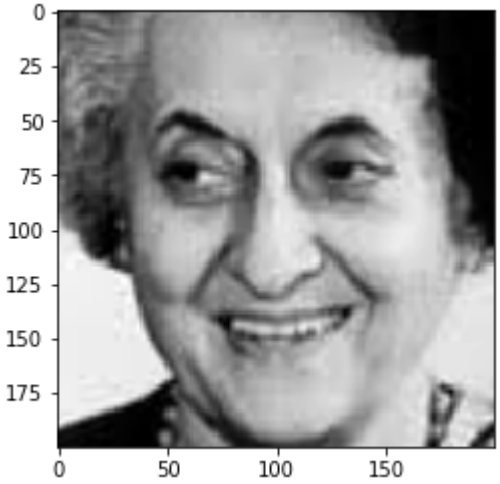

0
20
40
60
80
100
120
140
0
50
100
150
200
250
300


0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150
200
250

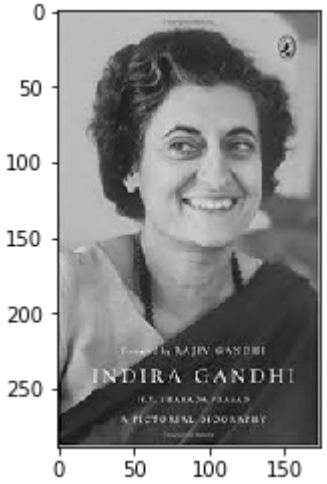

0
25
50
75
100
125
150
175
0
50
100
150

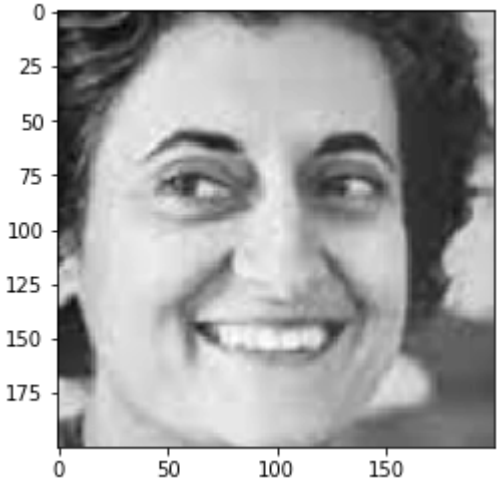

0
50
100
150
200
0
50
100

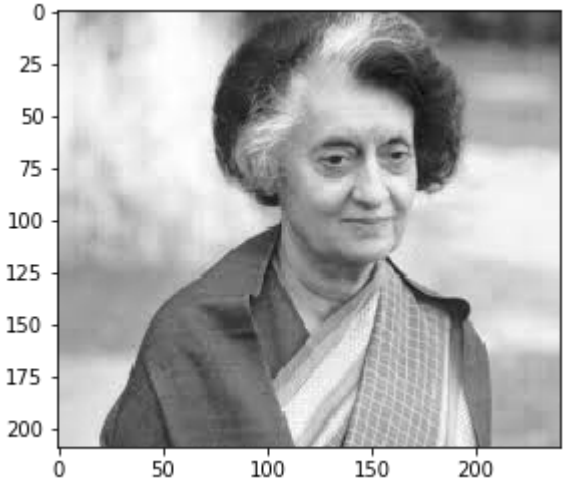

0
25
50
75
100
125
150
175
0
50
100
150

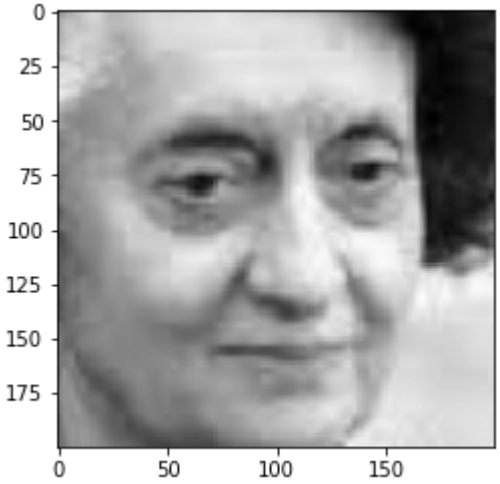

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

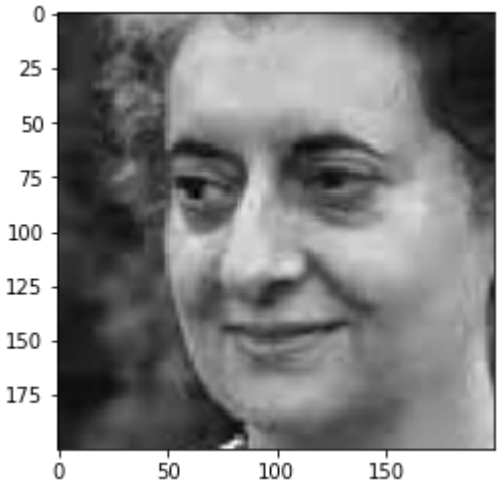

INDIRA GANDHI



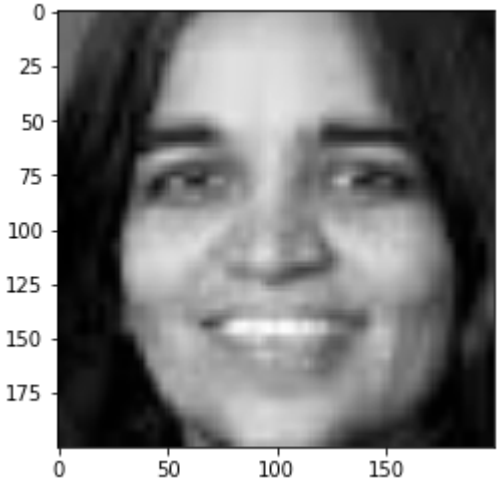



0
25
50
75
100
125
150
175
0
50
100
150

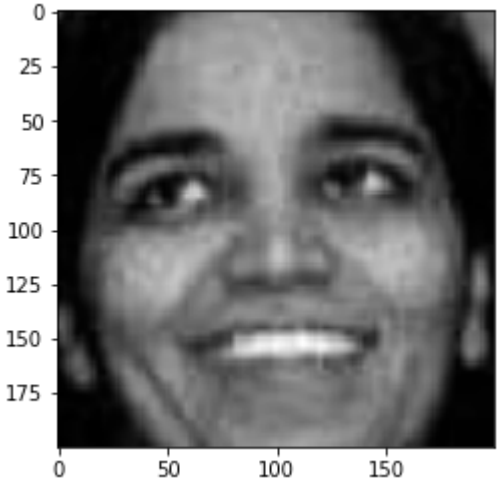

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

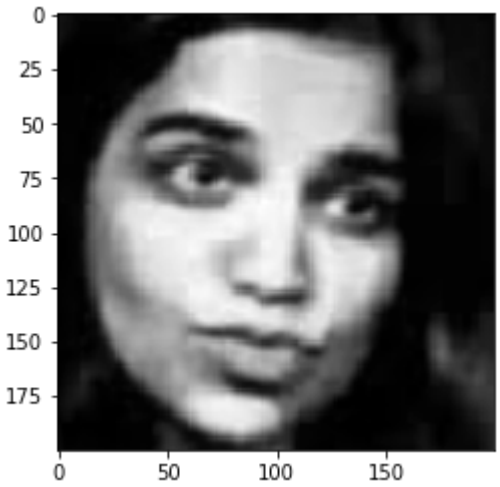

0
50
100
150
200
250
0
50
100
150
200

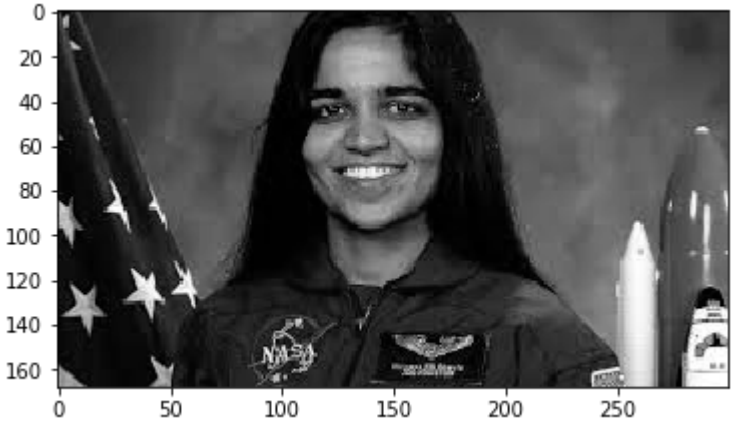

0
25
50
75
100
125
150
175
0
50
100
150

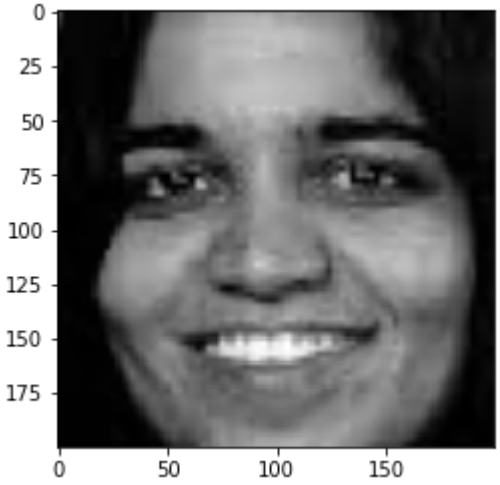

0
20
40
60
80
100
120
140
160
IMAGE TWEETED BY @rssurjewala
0
50
100
150
200


0
25
50
75
100
125
150
175
0
50
100
150

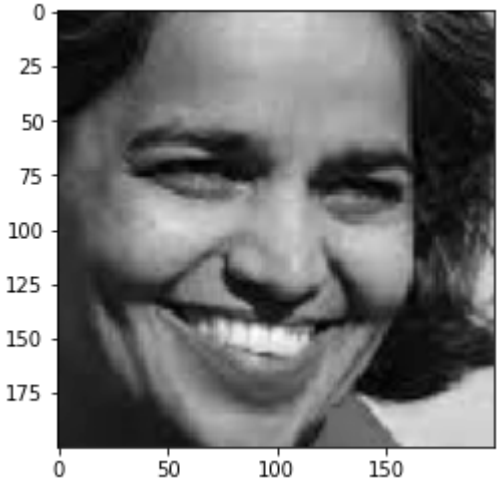

0
25
50
75
100
125
150
175
0
50
100
150
200
250

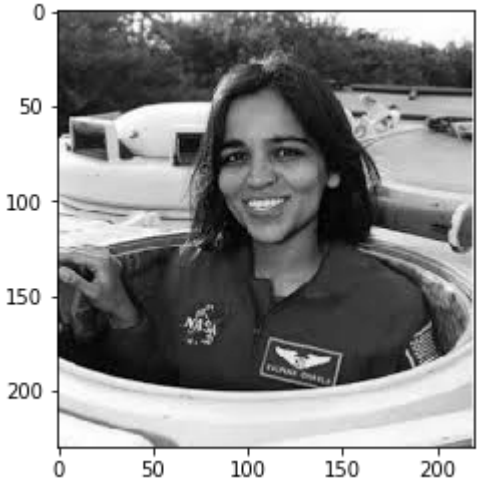

0
25
50
75
100
125
150
175
0
50
100
150

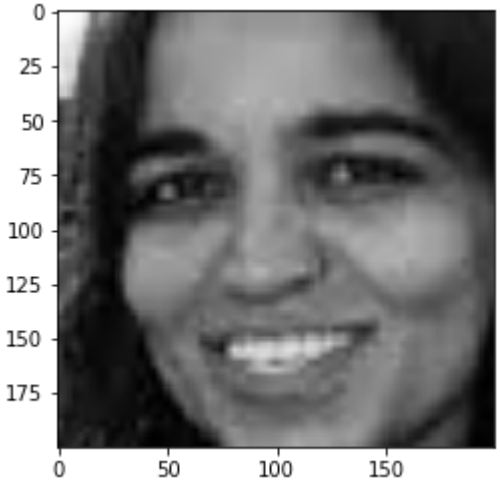

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250

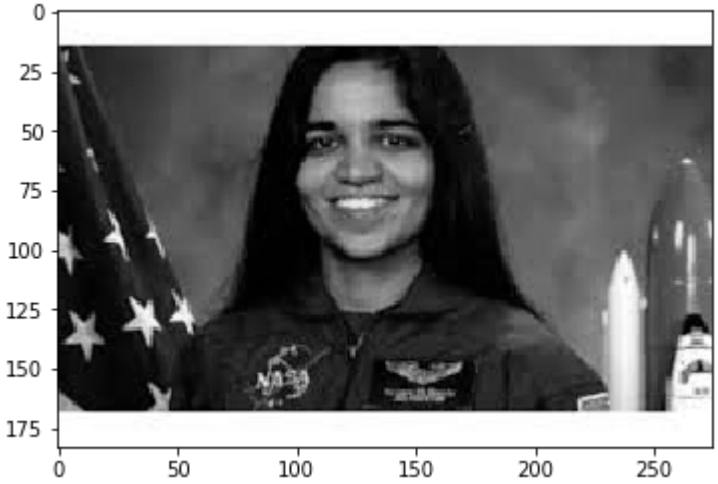

0
25
50
75
100
125
150
175
0
50
100
150

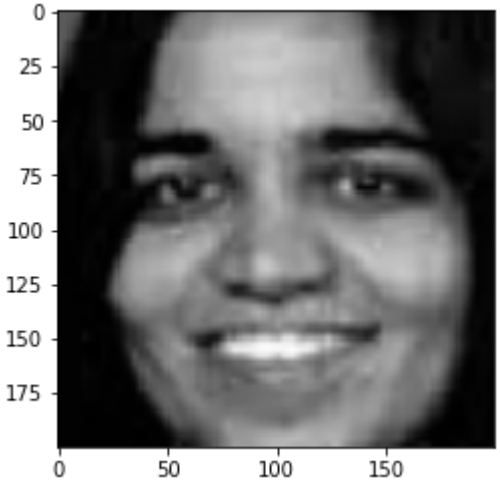

0
50
100
150
200
250
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150

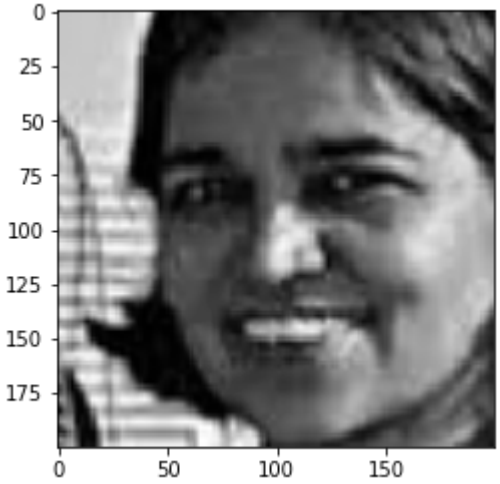

0
50
100
150
200
0
50
100
150
200

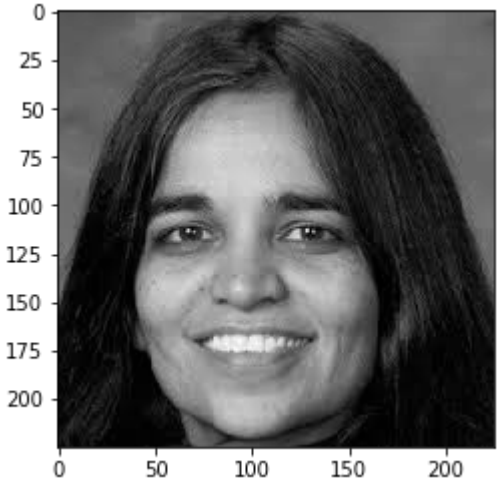

0
25
50
75
100
125
150
175
0
50
100
150

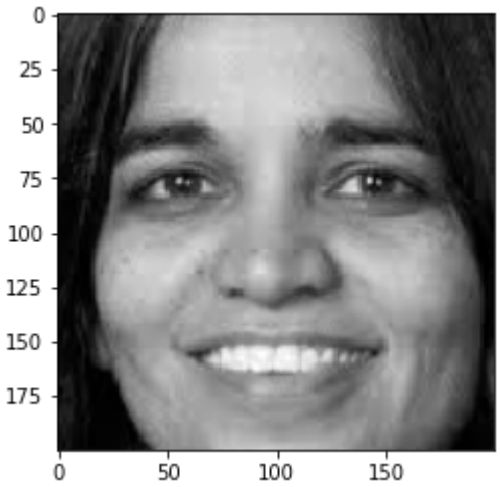

0
25
50
75
100
125
150
175
0
50
100
150
200
250

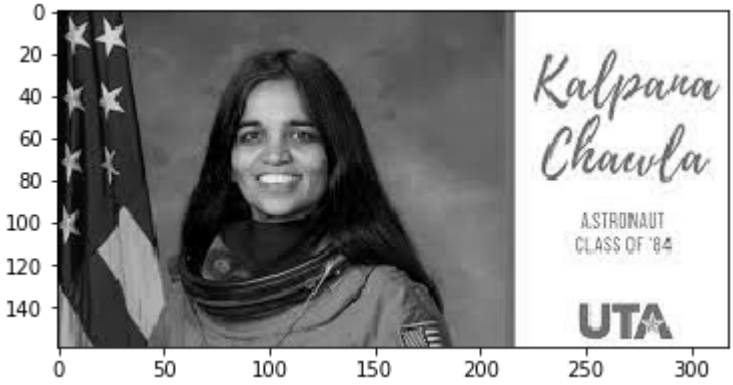

0
25
50
75
100
125
150
175
0
50
100
150

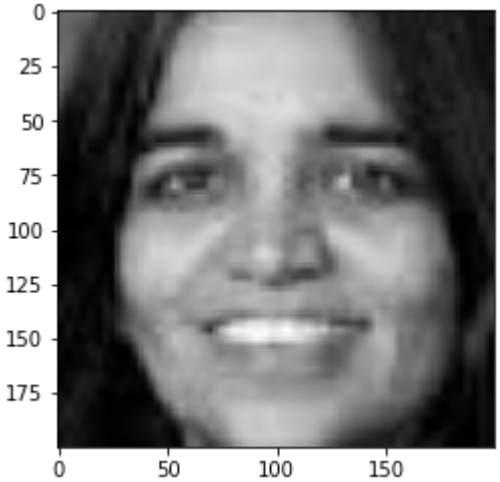

0
50
100
150
200
250
0
50
100
150

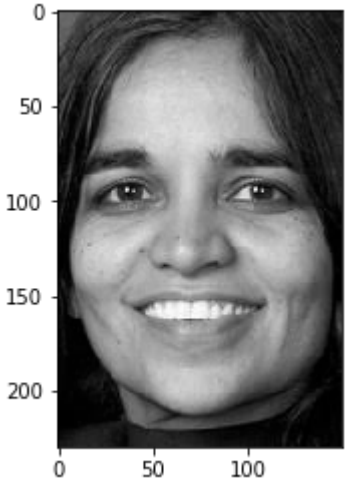

0
25
50
75
100
125
150
175
0
50
100
150

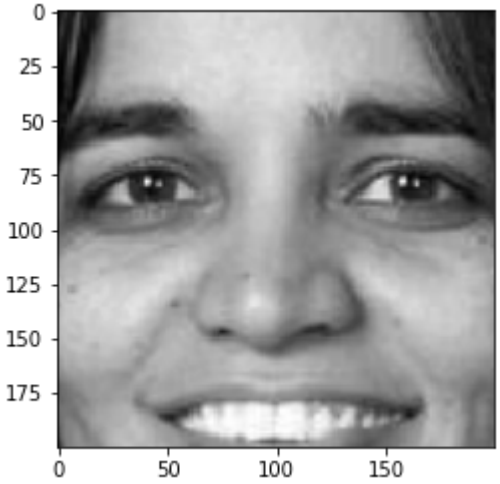

0
25
50
75
100
125
150
175
200
0
50
100
150
200

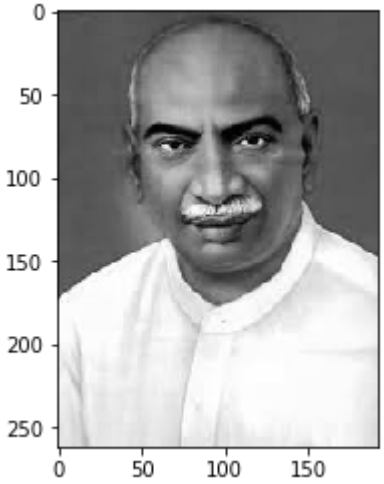

0
25
50
75
100
125
150
175
0
50
100
150

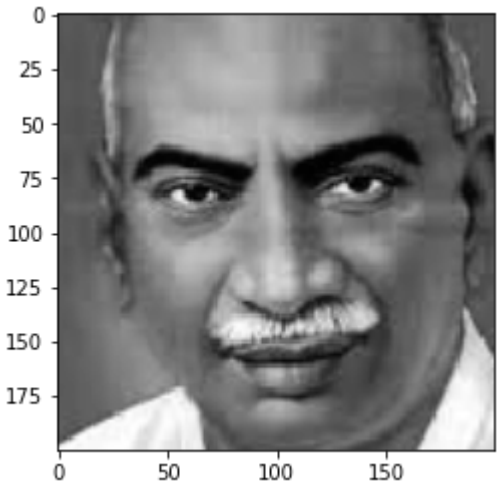

Kalpana
Chawla
ASTRONAUT
CLASS OF '84
UTA






0
25
50
75
100
125
150
175
0
50
100
150


0
50
100
150
200
250
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150

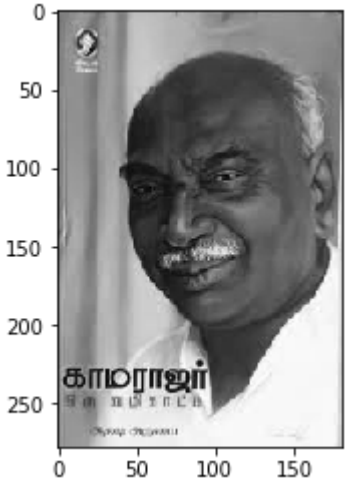


Image not good kmj10.jpg

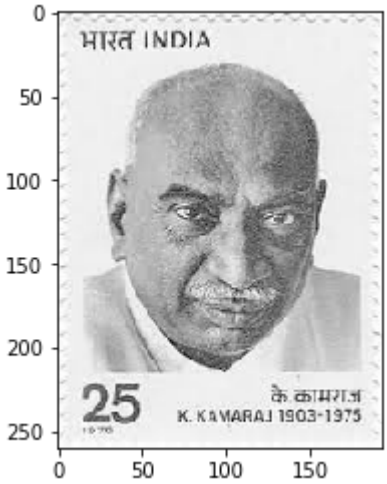
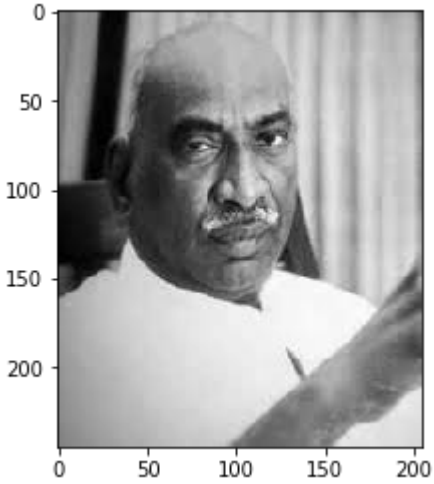
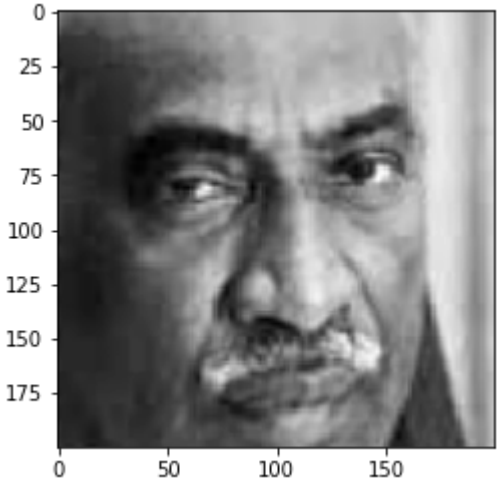


Image not good kmj11.jpg





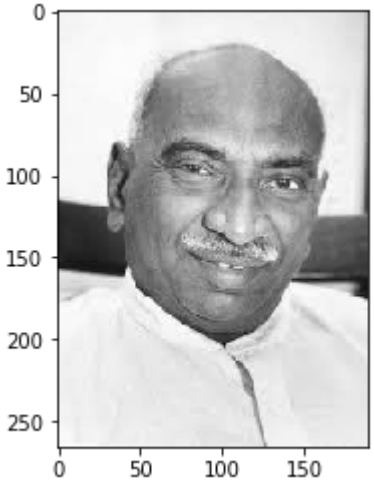
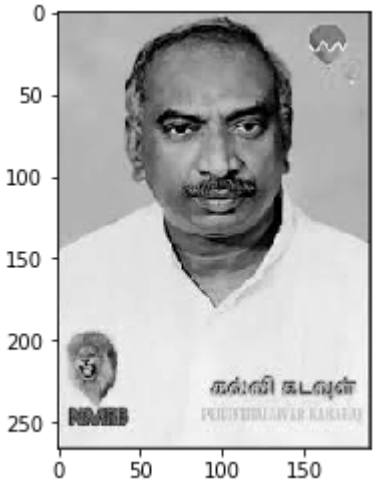
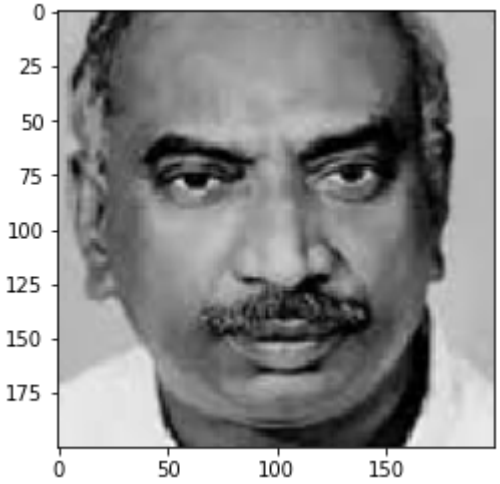
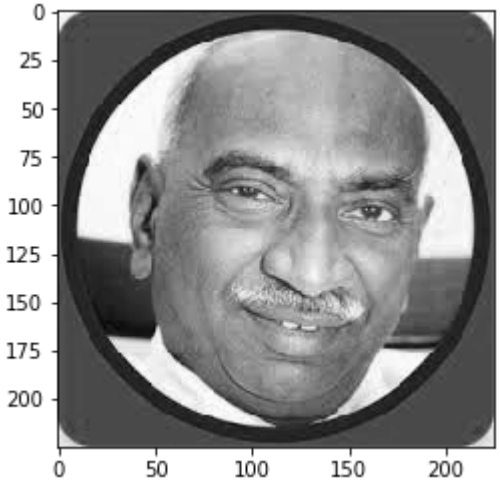
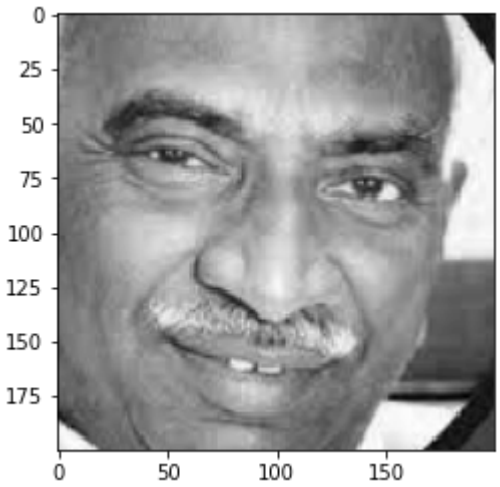


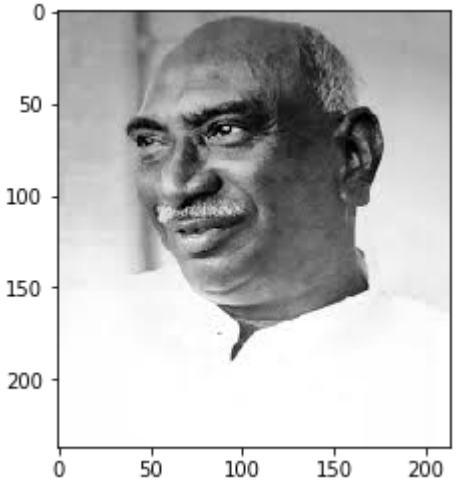
Image not good kmj2.jpg



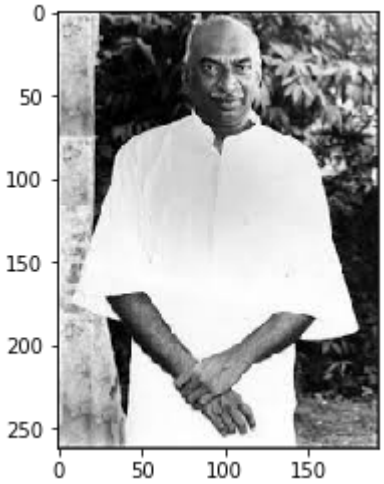

0
25
50
75
100
125
150
175
0
50
100
150

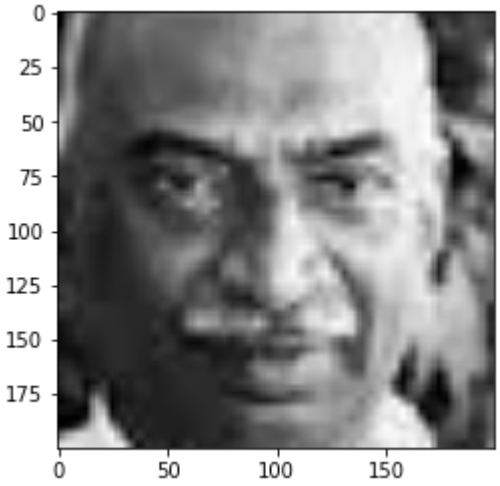

0
25
50
75
100
125
150
175
200
0
50
100
150
200

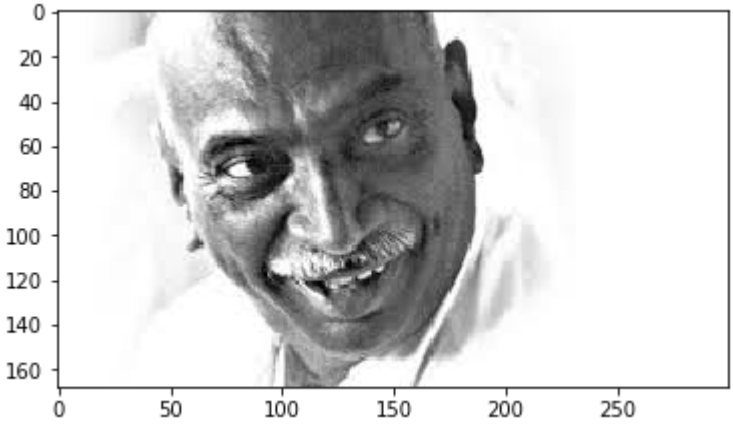

0
25
50
75
100
125
150
175
0
50
100
150



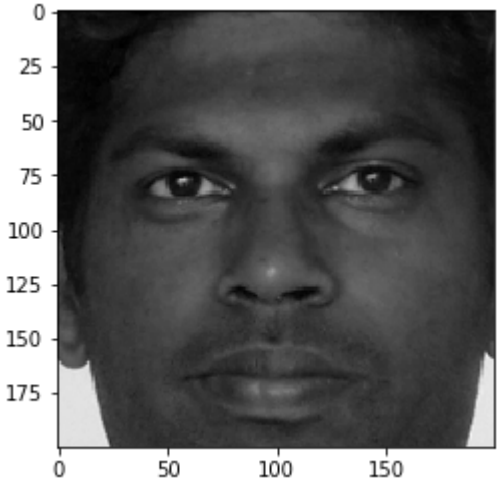

```
Image not good kmj6.jpg
```

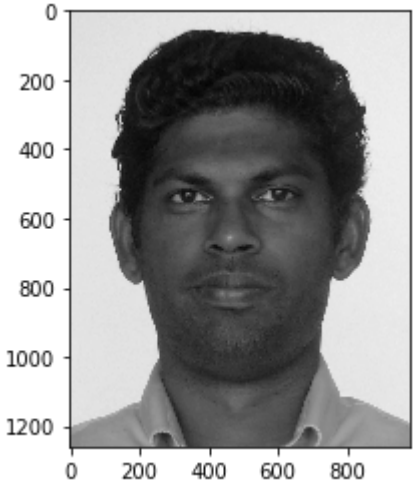


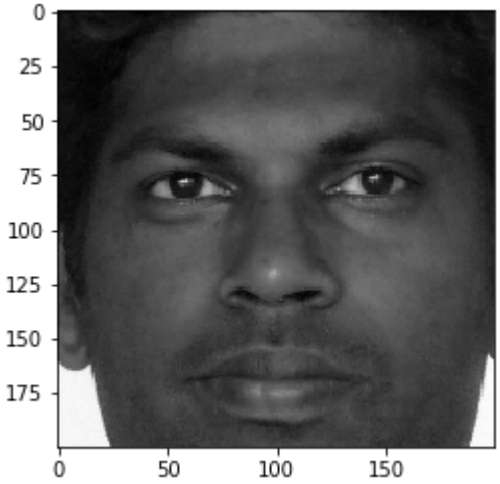


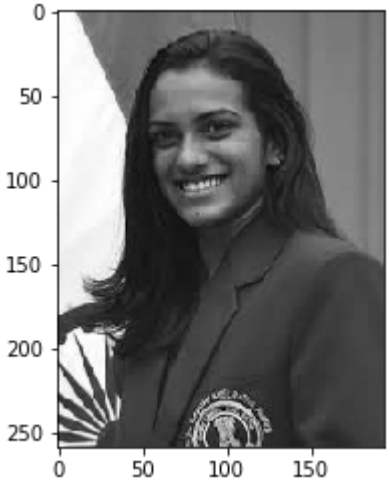


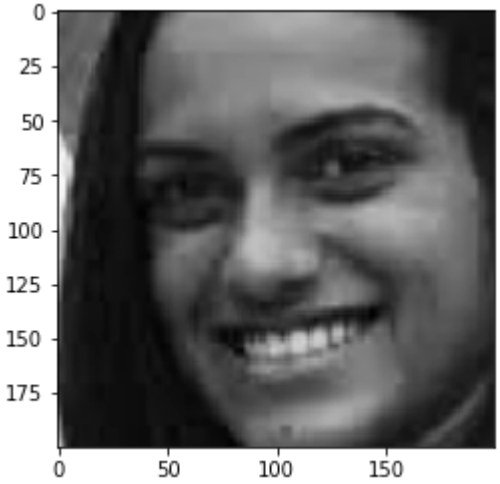

```
Image not good kmj8.jpg
```

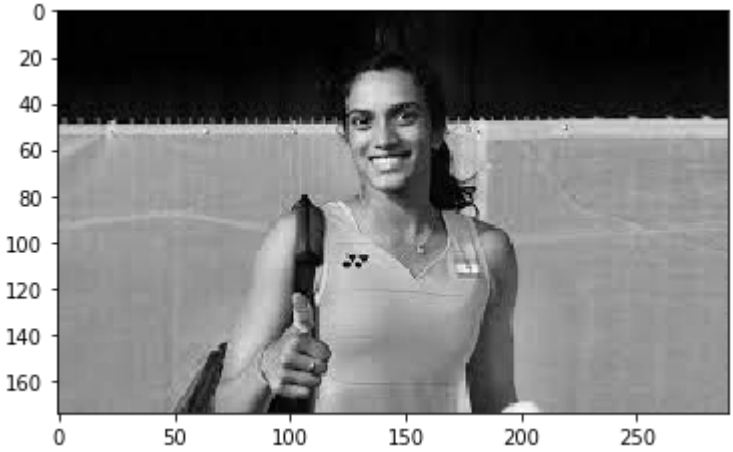


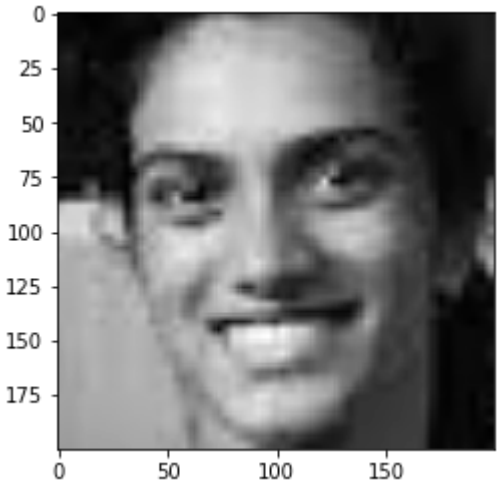

0
200
400
600
800
1000
1200
0
200
400
600
800


0
25
50
75
100
125
150
175
0
50
100
150

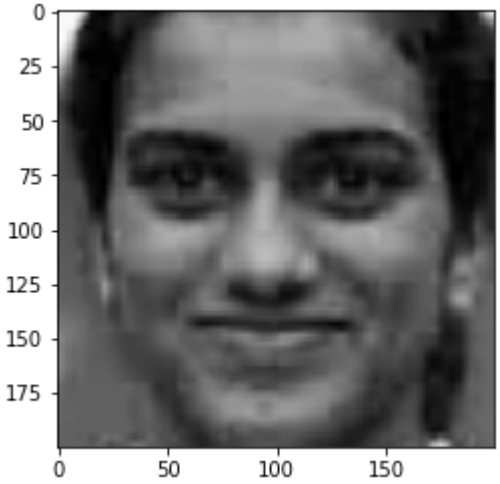

0
50
100
150
200
250
0
50
100
150

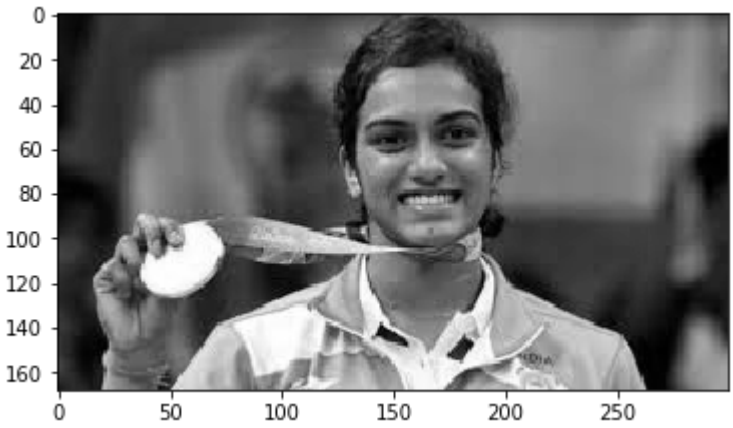

0
25
50
75
100
125
150
175
0
50
100
150

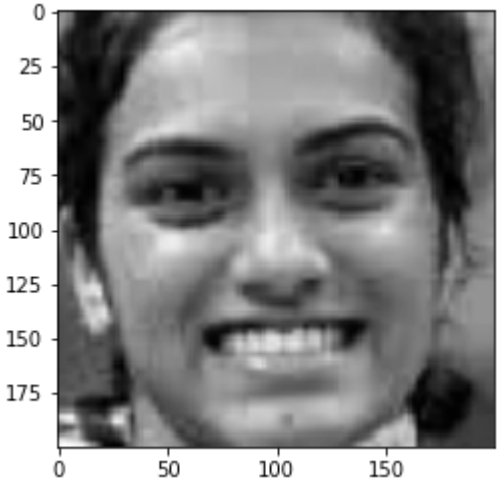

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

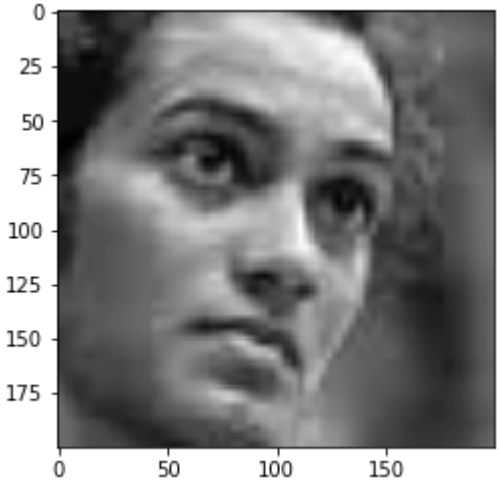

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

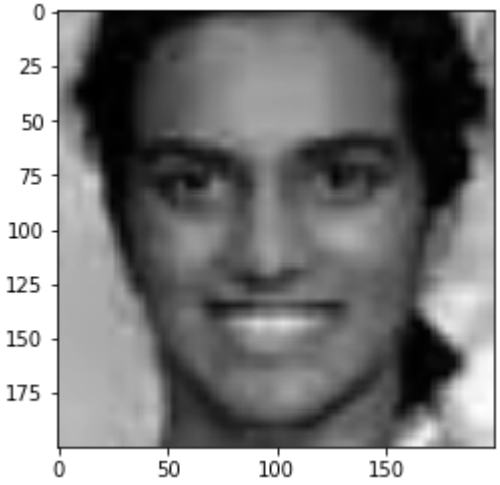

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250

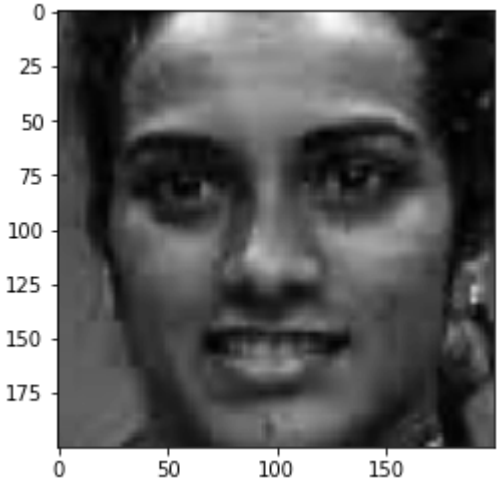

0
25
50
75
100
125
150
175
0
50
100
150


0
20
40
60
80
100
120
140
160
0
50
100
150
200
250

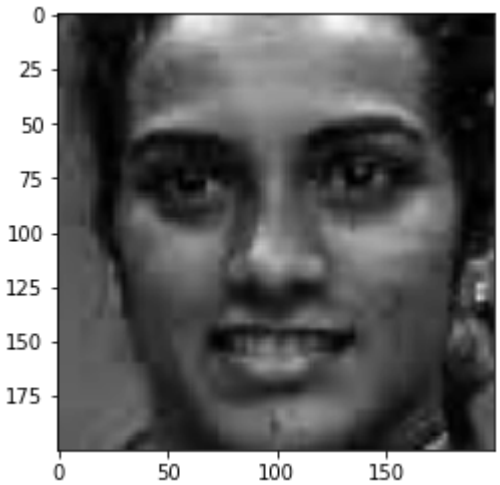

0
25
50
75
100
125
150
175
0
50
100
150

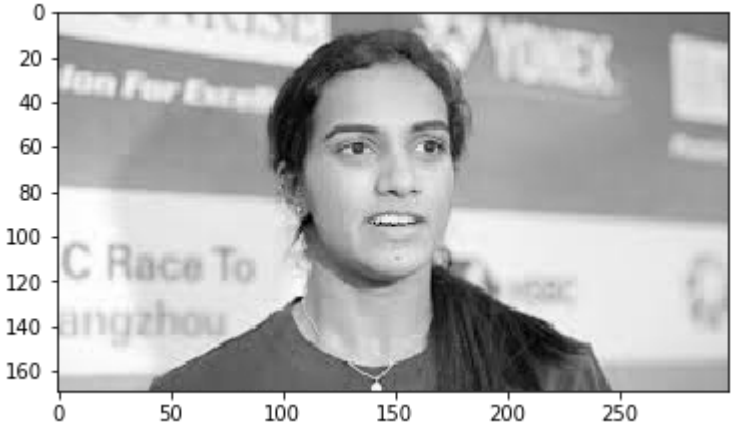

pnb MetLife
YONEX

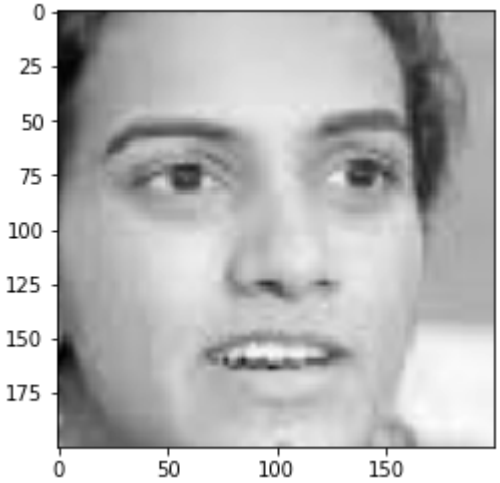



0
25
50
75
100
125
150
175
0
50
100
150

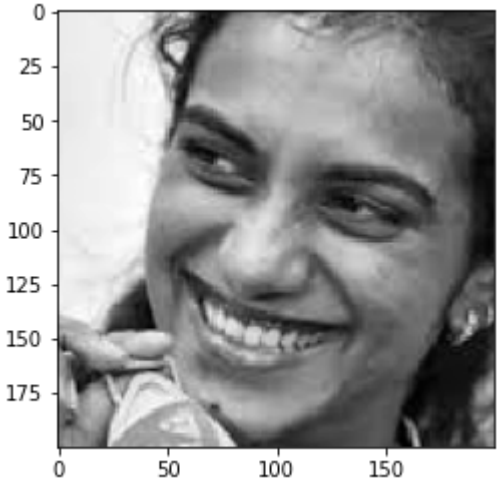

0
25
50
75
100
125
150
175
0
50
100
150
200
250

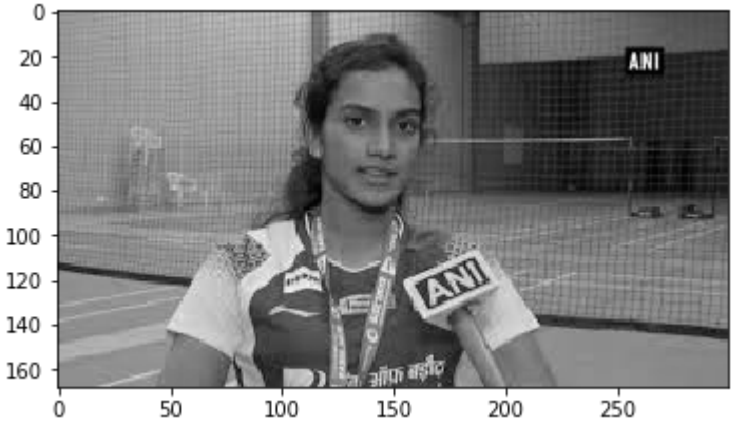

0
25
50
75
100
125
150
175
0
50
100
150

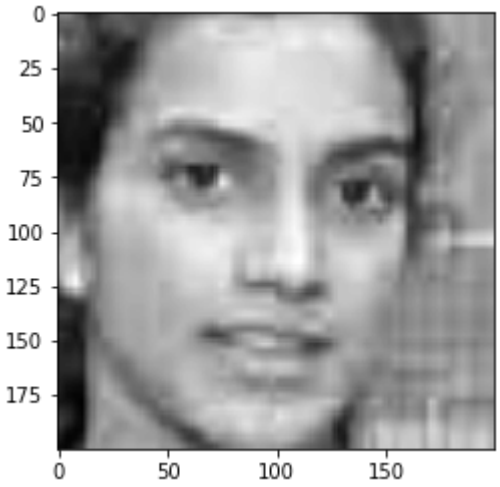

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

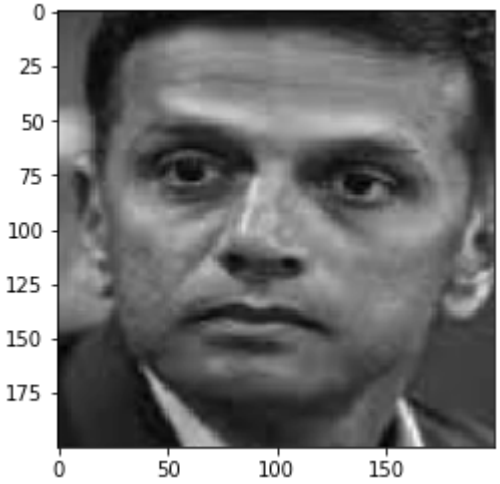

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

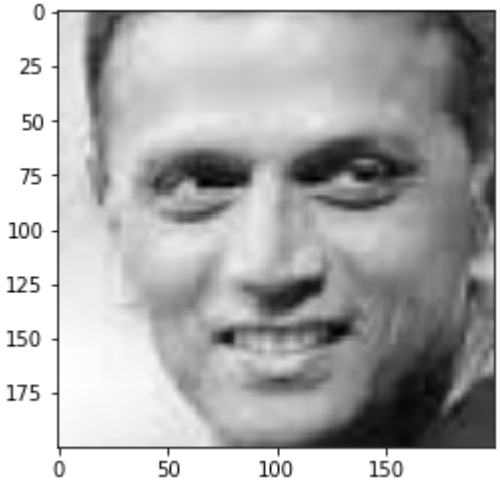

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250
ANI
ANI


0
25
50
75
100
125
150
175
0
50
100
150

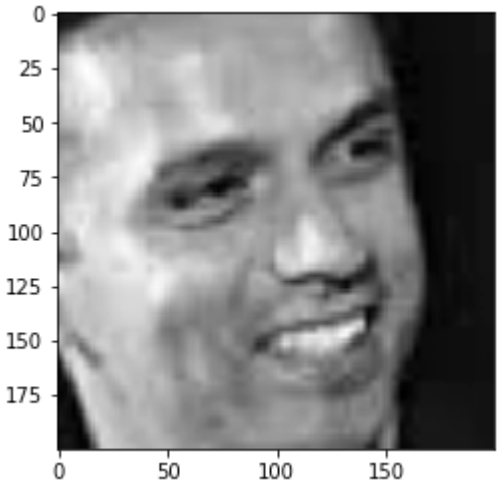

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

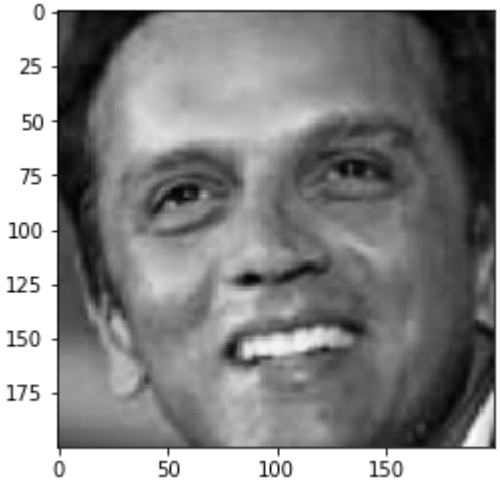

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

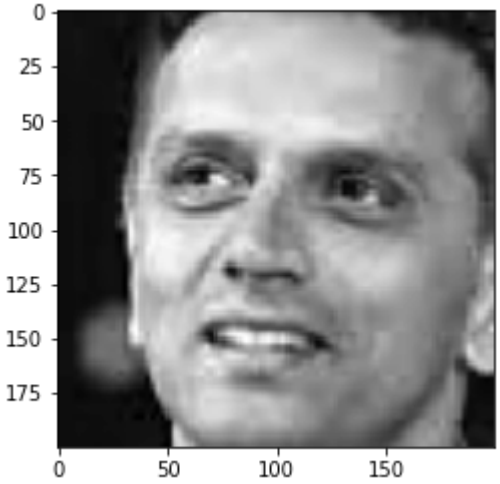

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

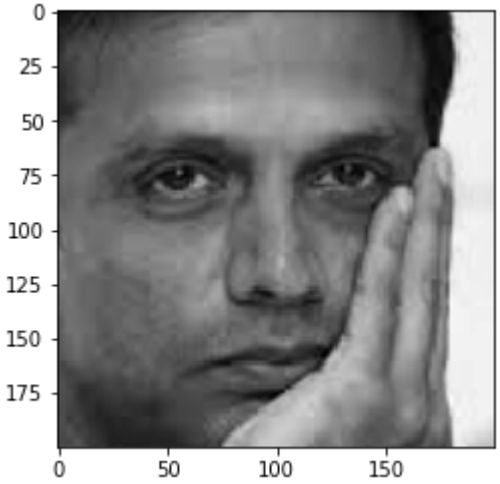

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

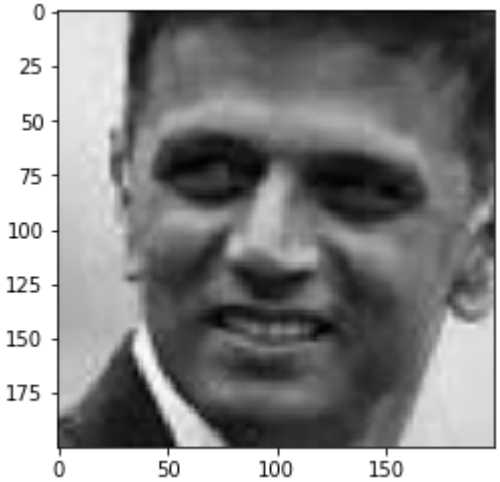

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

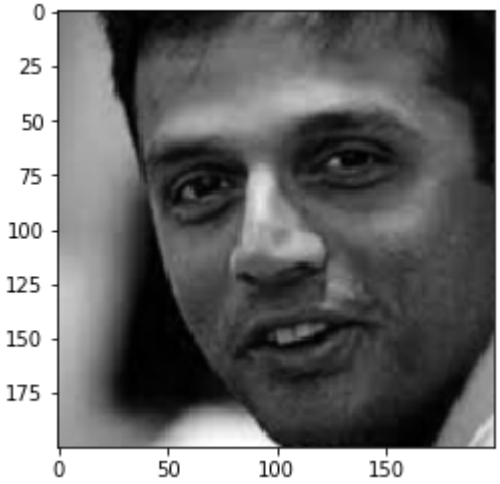

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

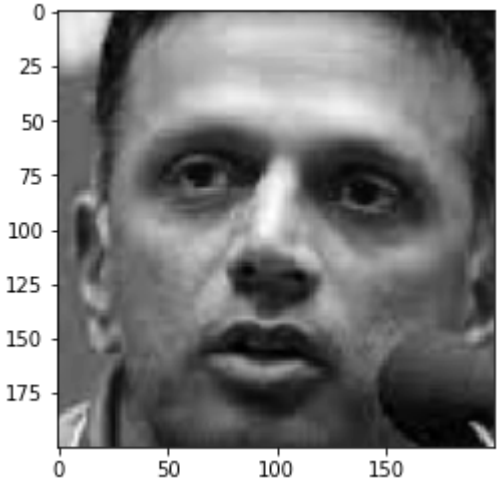

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250

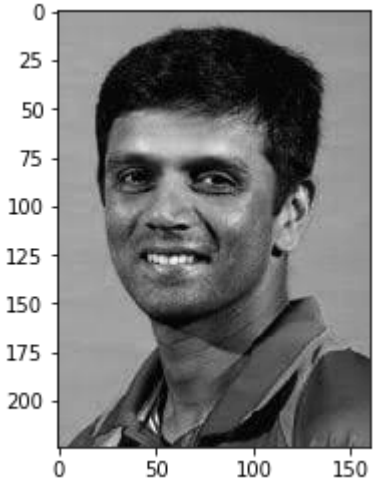

0
25
50
75
100
125
150
175
0
50
100
150

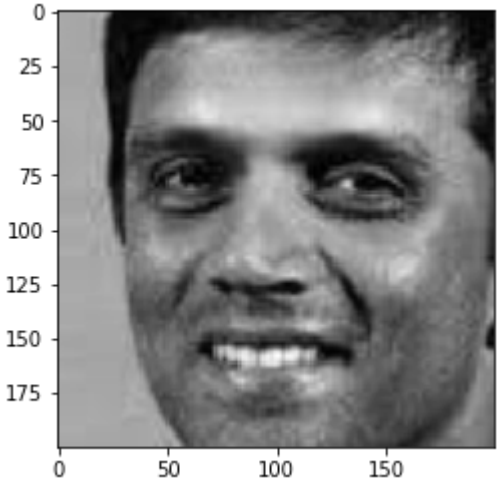

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150


0
20
40
60
80
100
120
140
0
50
100
150
200
250
300


0
25
50
75
100
125
150
175
0
50
100
150





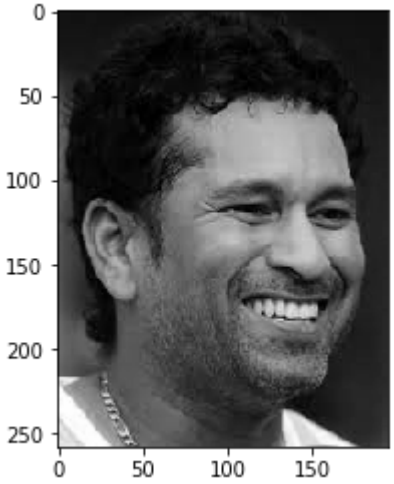
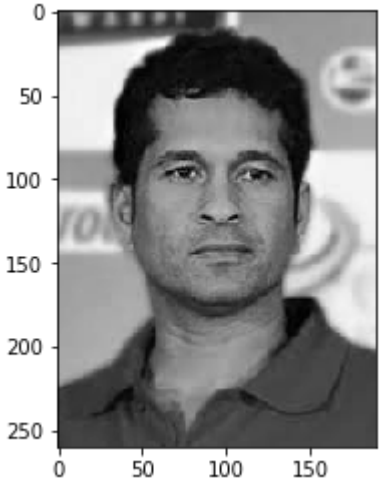
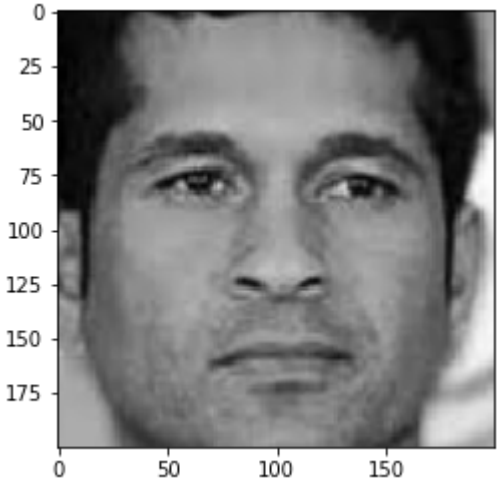
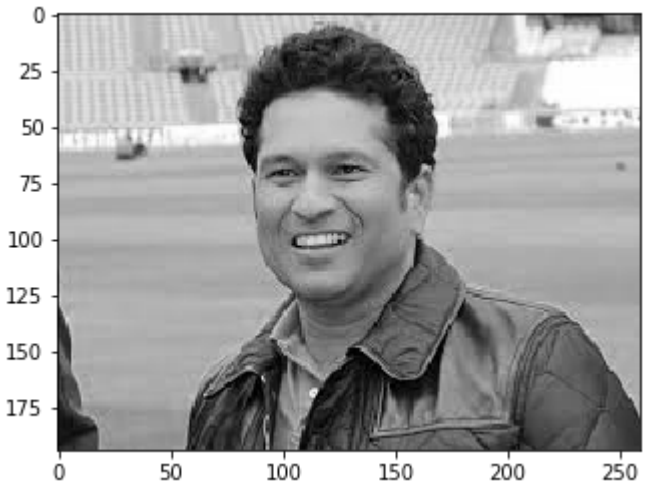
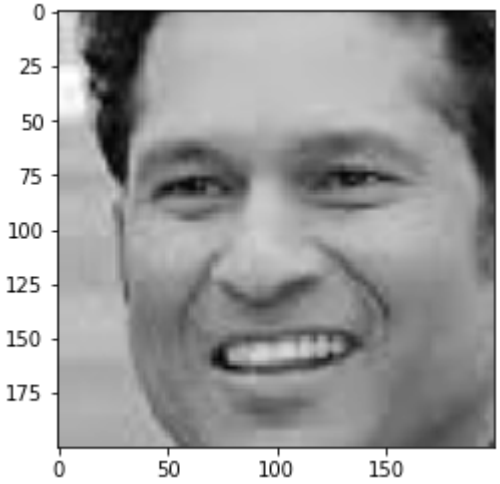


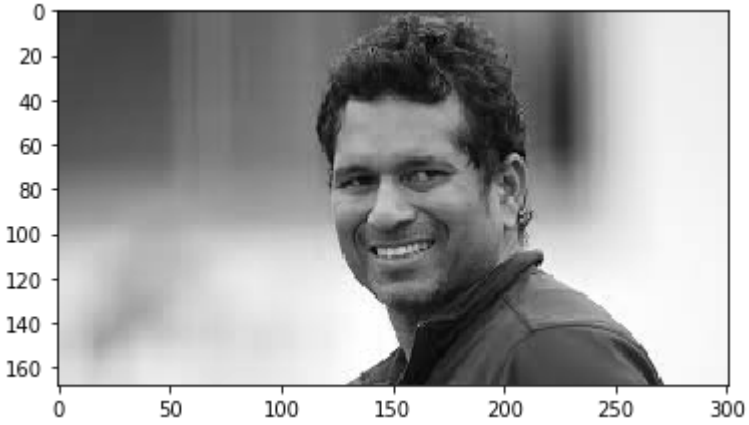
Image not good sc1.jfif

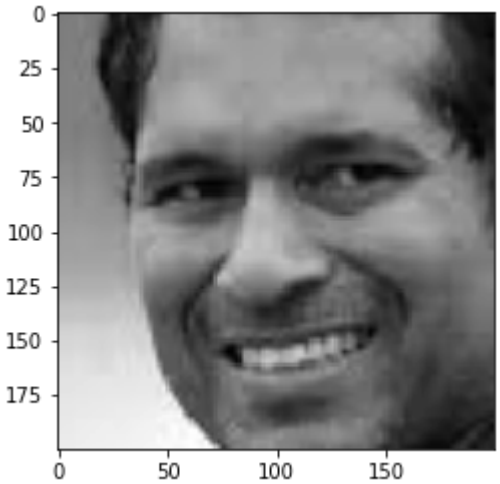

0
50
100
150
200
250
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150

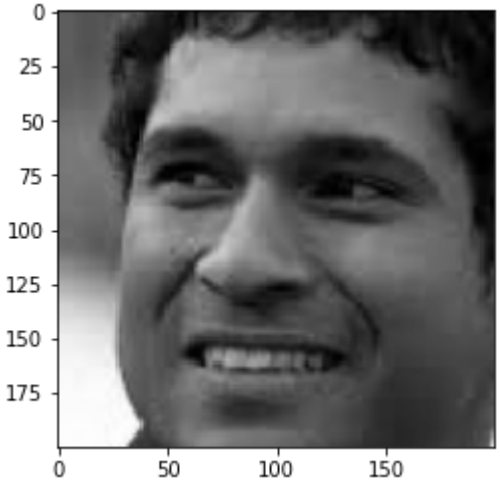

0
25
50
75
100
125
150
175
0
50
100
150
200
250

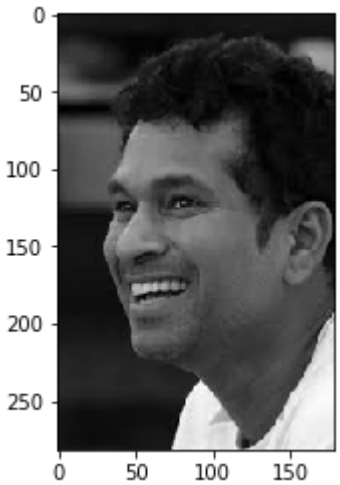

0
25
50
75
100
125
150
175
0
50
100
150

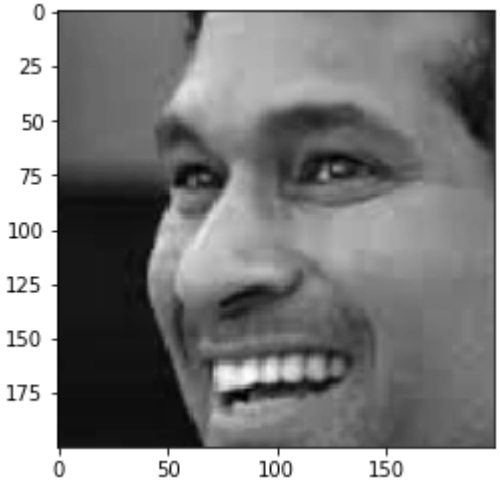

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250
300

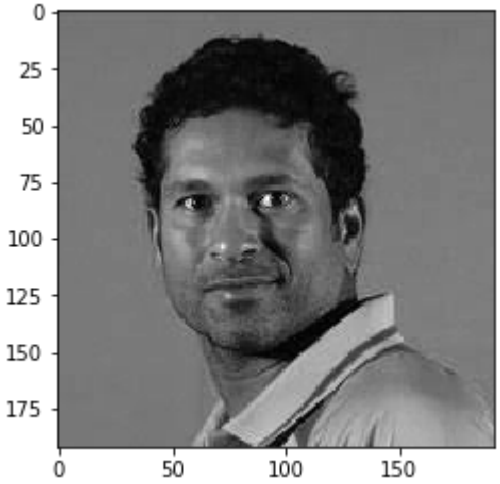

0
25
50
75
100
125
150
175
0
50
100
150

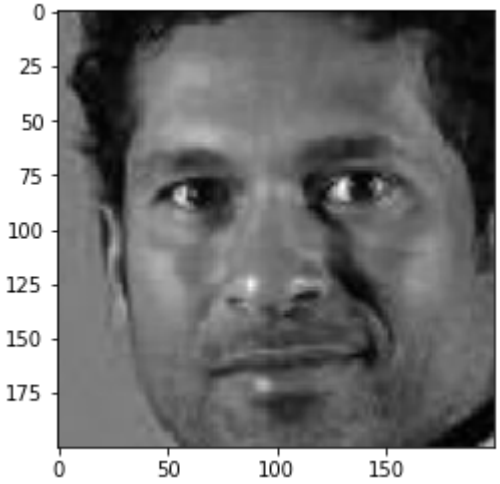

0
25
50
75
100
125
150
175
0
50
100
150
200
250

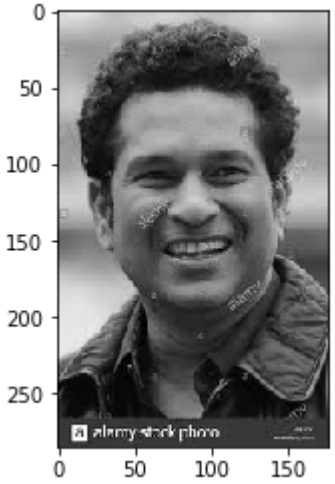

0
25
50
75
100
125
150
175
0
50
100
150

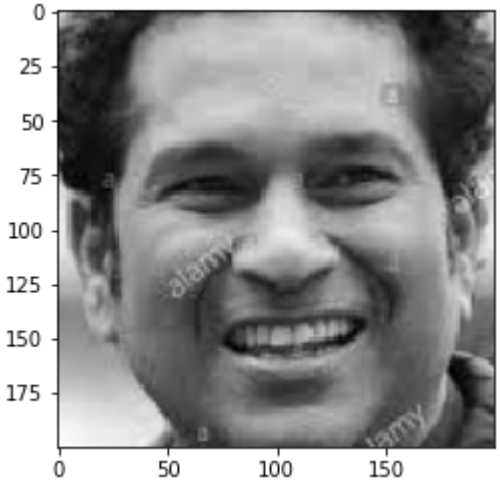

0
50
100
150
200
250
0
50
100
150

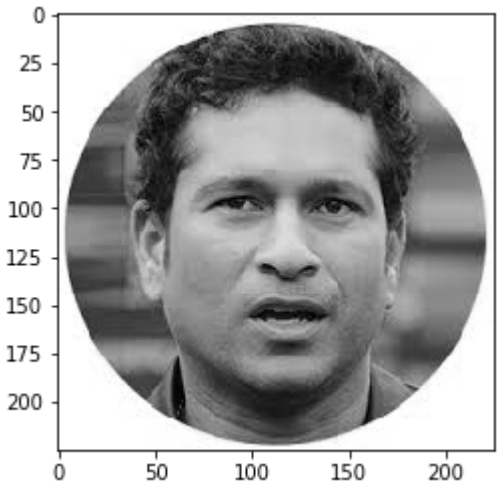

0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150


0
50
100
150
200
250
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
200
0
50
100
150
200

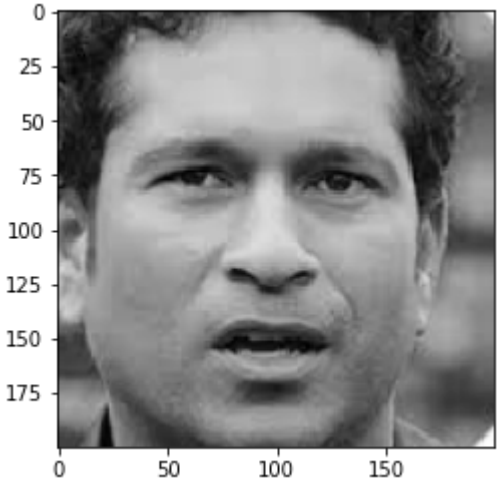
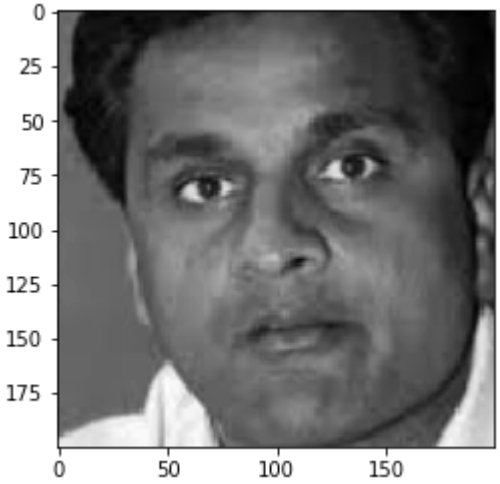
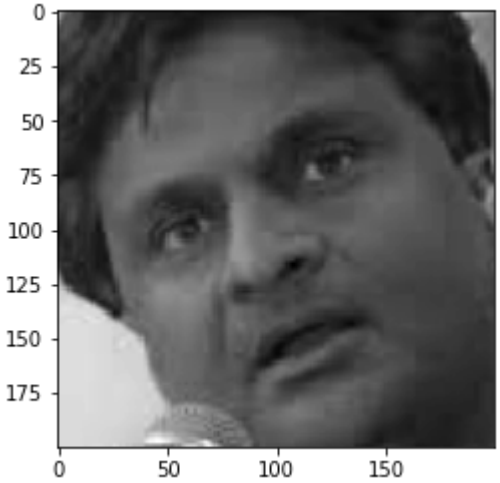


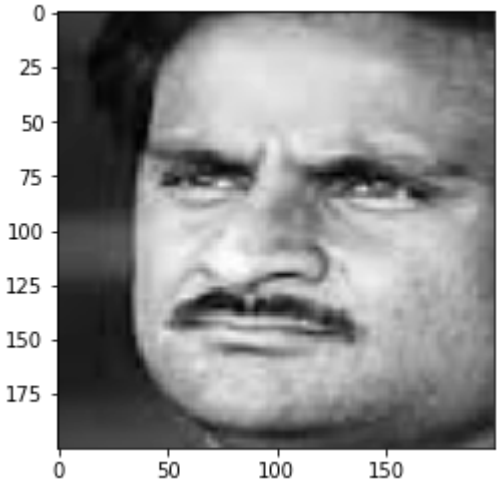
Image not good sc9.jfif


0
25
50
75
100
125
150
175
0
50
100
150

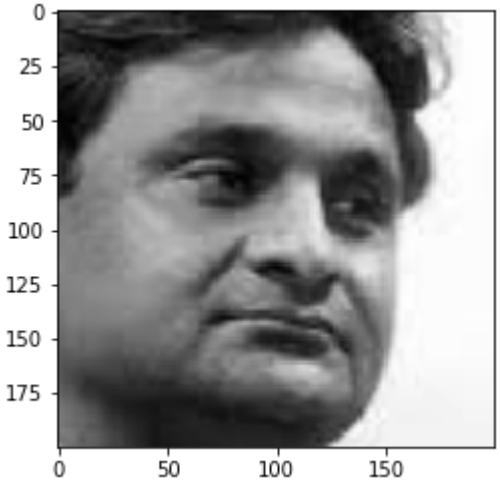

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

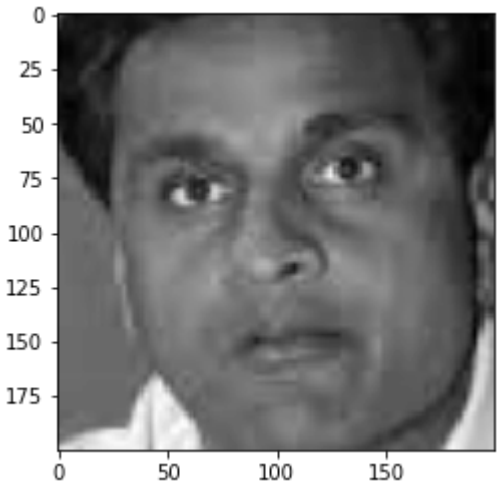

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250

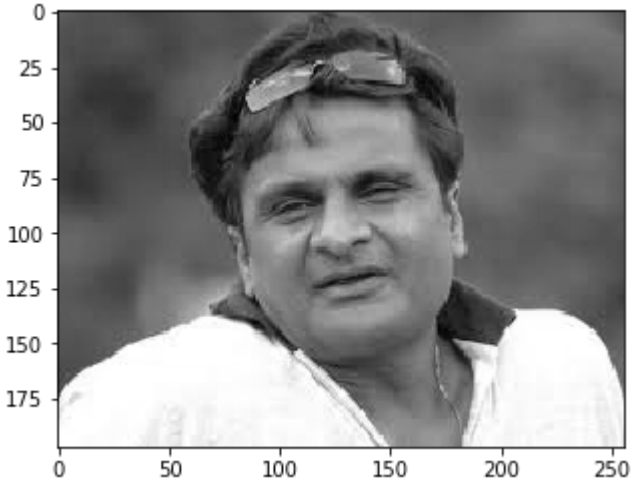

0
25
50
75
100
125
150
175
0
50
100
150

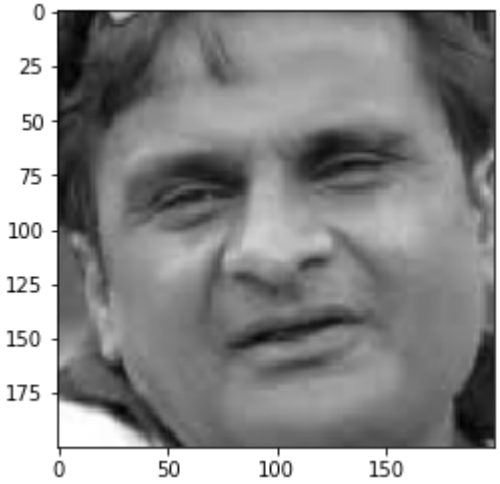

0
25
50
75
100
125
150
175
0
50
100
150
200
250

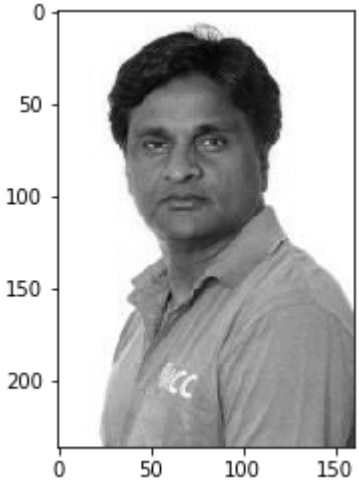

0
25
50
75
100
125
150
175
0
50
100
150

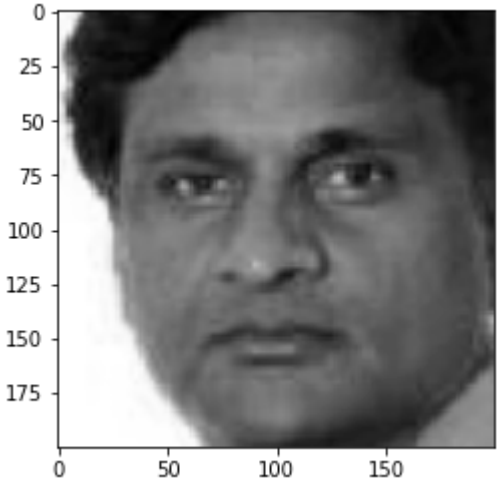

0
50
100
150
200
250
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150

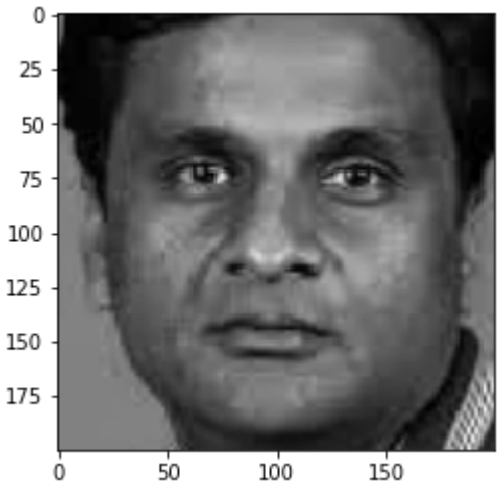

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

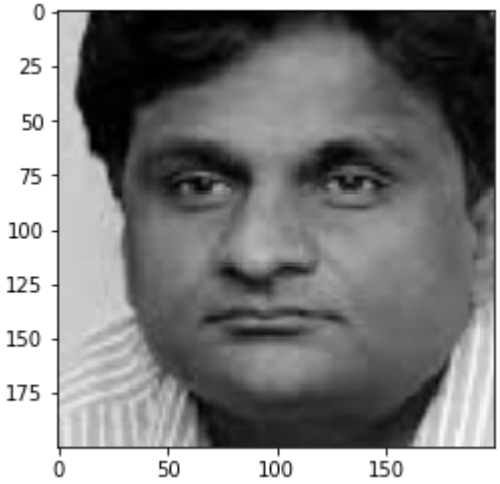

0
50
100
150
200
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150

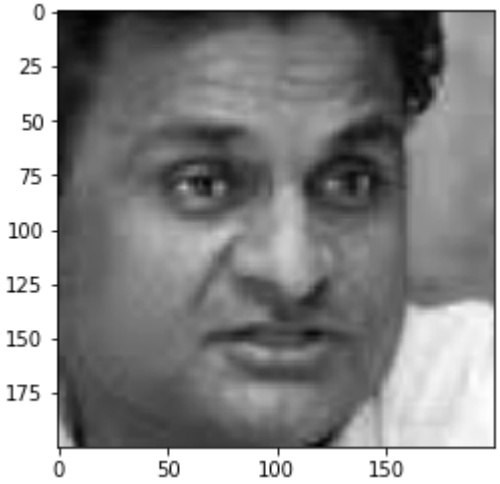

0
50
100
150
200
250
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150

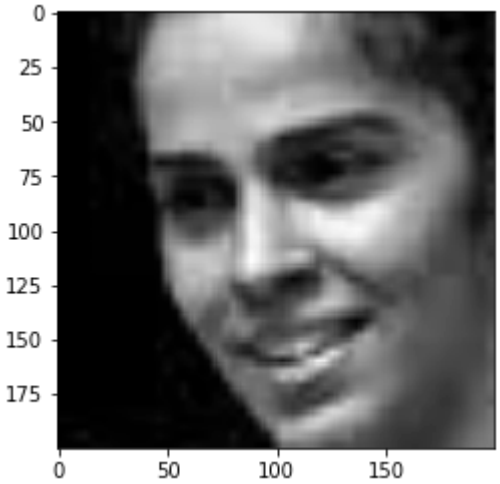

0
25
50
75
100
125
150
175
0
50
100
150
200
250

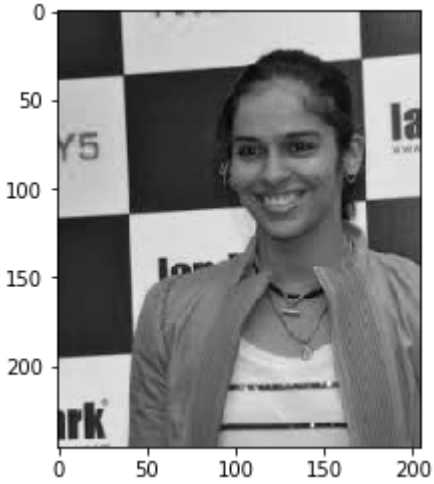

0
25
50
75
100
125
150
175
0
50
100
150

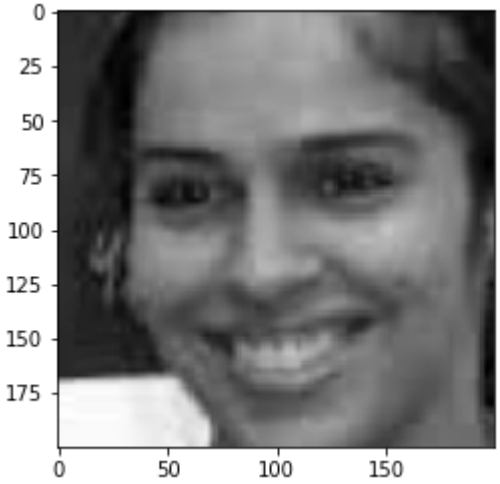

0
25
50
75
100
125
150
175
0
50
100
150
200
250

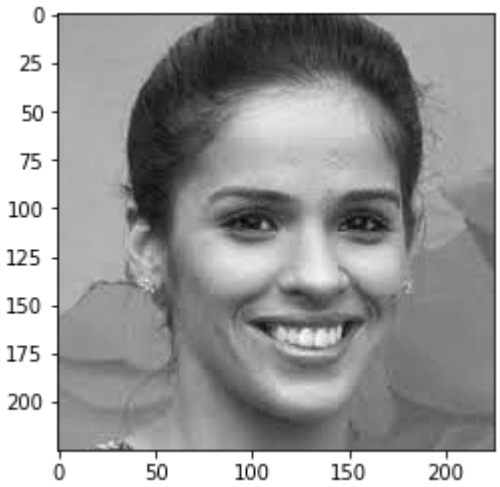

0
25
50
75
100
125
150
175
0
50
100
150

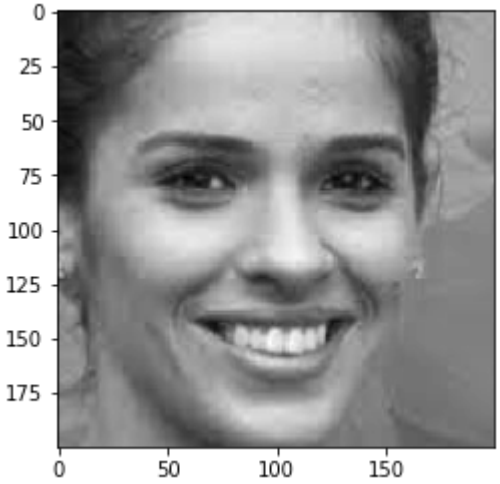

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

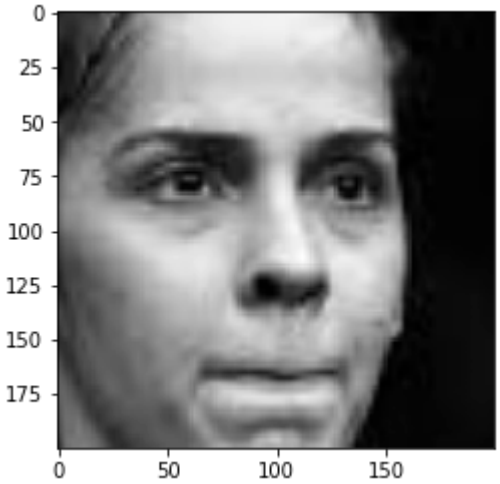

0
50
100
150
200
0
50
100
150
200


0
25
50
75
100
125
150
175
0
50
100
150

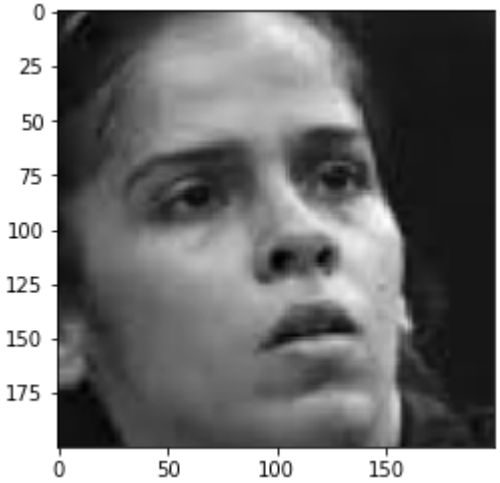

0
25
50
75
100
125
150
175
200
0
50
100
150
200

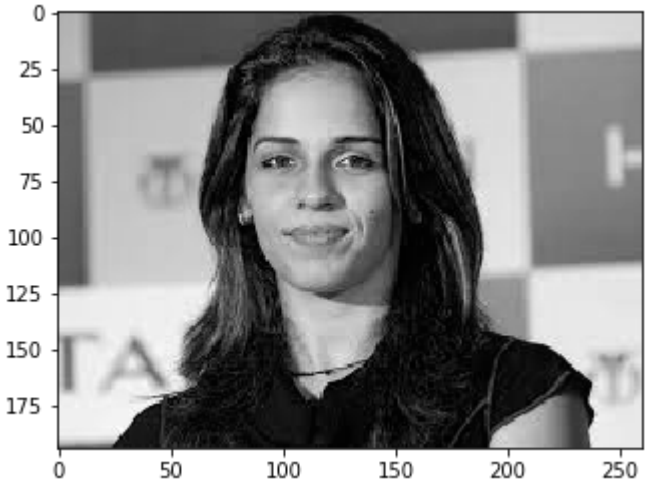

0
25
50
75
100
125
150
175
0
50
100
150

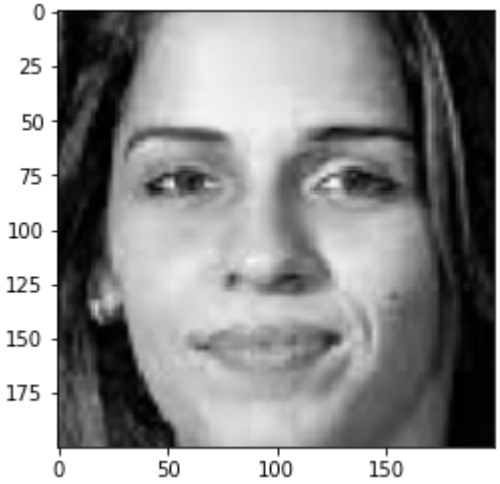

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250

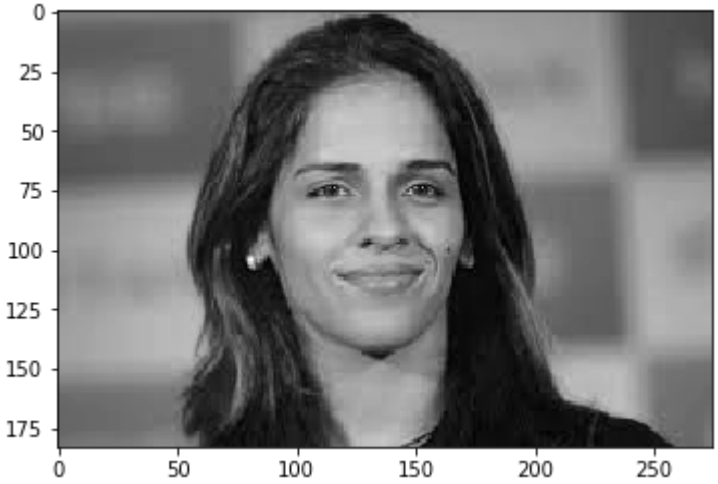

0
25
50
75
100
125
150
175
0
50
100
150

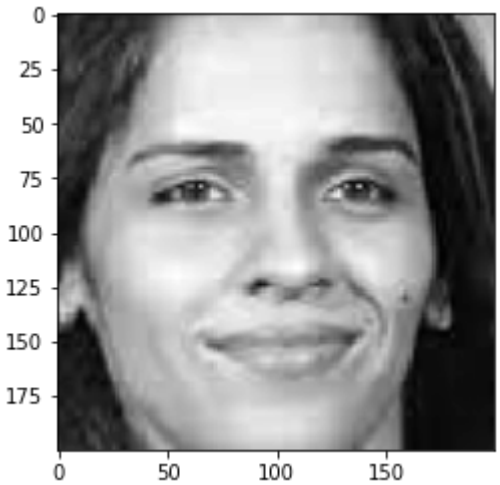

0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

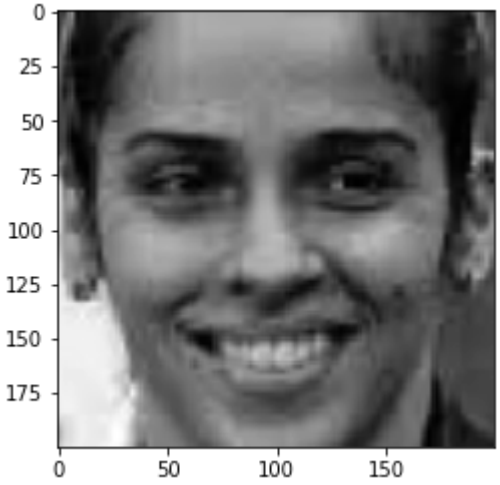

0
25
50
75
100
125
150
175
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150

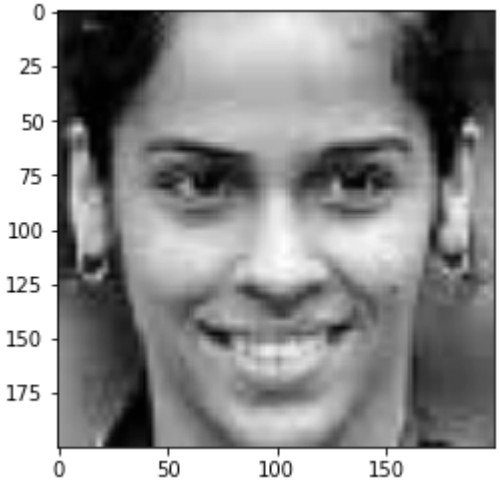

0
25
50
75
100
125
150
175
0
50
100
150
200
250

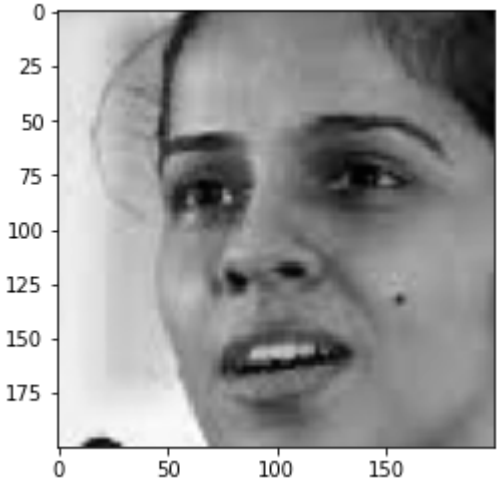

0
25
50
75
100
125
150
175
0
50
100
150


0
20
40
60
80
100
120
140
160
0
50
100
150
200
250


0
25
50
75
100
125
150
175
0
50
100
150


0
20
40
60
80
100
120
140
160
0
50
100
150
200
250





In [6]:
```
# Standardizing the Observations
M=pd.DataFrame(dm,columns =fname.tolist())
Mmean = M.mean(axis=1)
NM=M.sub(Mmean,axis='rows')
pca = PCA(n_components=ncompnts)
NMtransform = pca.fit_transform(NM.T)
NMtransform .shape
```

Out[6]: (89, 20)

In [7]:
```
#Eigenfaces

principalDf = pd.DataFrame(data = pca.components_.T)
principalDf
```

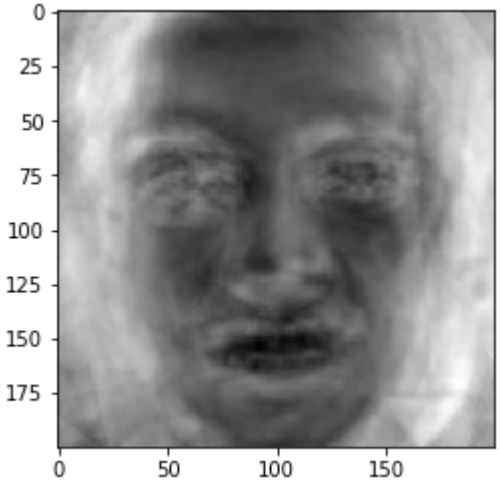
Out[7]:

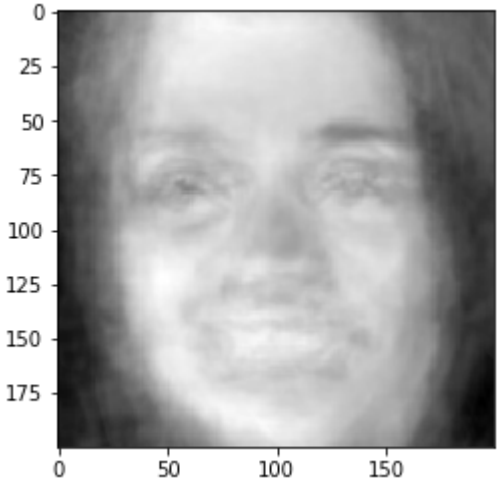
	0	1	2	3	4	5	6	7	
0	0.005283	-0.005614	-0.003565	0.005432	0.002135	-0.002258	-0.004209	-0.008565	-0.0
1	0.005397	-0.005730	-0.003560	0.005695	0.001875	-0.002078	-0.003871	-0.008811	-0.0
2	0.005677	-0.006141	-0.003371	0.006080	0.001785	-0.001581	-0.003016	-0.009515	-0.0
3	0.005856	-0.006584	-0.002972	0.006258	0.002187	-0.001327	-0.002078	-0.010278	-0.0
4	0.005881	-0.006733	-0.002704	0.006601	0.002433	-0.000538	-0.001055	-0.010294	-0.0
5	0.005747	-0.006858	-0.002827	0.006846	0.002643	-0.000384	-0.000591	-0.010297	-0.0
6	0.005715	-0.006611	-0.003330	0.007196	0.002855	-0.000092	-0.000273	-0.010264	-0.0
7	0.005706	-0.005990	-0.003427	0.007719	0.002752	0.000791	-0.000290	-0.009969	-0.0
8	0.005857	-0.005389	-0.003337	0.008392	0.002725	0.001474	0.000295	-0.009558	-0.0
9	0.005906	-0.005116	-0.003069	0.008858	0.002495	0.001743	0.001137	-0.008957	-0.0
10	0.005749	-0.005109	-0.002909	0.009469	0.001788	0.001939	0.002029	-0.008973	-0.0
11	0.005717	-0.004994	-0.002476	0.010310	0.001099	0.001965	0.003063	-0.008924	-0.0
12	0.005831	-0.004798	-0.001893	0.011261	0.000898	0.001976	0.003777	-0.008298	-0.0
13	0.006236	-0.004483	-0.001411	0.011894	0.000589	0.001880	0.004008	-0.007666	-0.0
14	0.006651	-0.004300	-0.001154	0.012415	0.000017	0.001899	0.004183	-0.007245	-0.0
15	0.006842	-0.004049	-0.000911	0.013060	-0.000847	0.001427	0.004669	-0.005800	-0.0
16	0.006943	-0.003886	-0.000695	0.013545	-0.001706	0.000703	0.005015	-0.004039	-0.0
17	0.006940	-0.003782	-0.000588	0.013997	-0.002110	0.000529	0.005038	-0.002704	-0.0
18	0.006948	-0.003167	-0.000348	0.014690	-0.002164	0.000461	0.004624	-0.000996	-0.0
19	0.006982	-0.002203	-0.000094	0.015428	-0.001553	0.001100	0.003685	0.000264	-0.0
20	0.006955	-0.001494	-0.000122	0.015655	-0.001095	0.001491	0.003751	0.001061	-0.0
21	0.006760	-0.001223	-0.000371	0.015777	-0.000973	0.001402	0.004270	0.001349	-0.0
22	0.006871	-0.001083	-0.000567	0.015639	-0.001049	0.000999	0.005163	0.001266	-0.0
23	0.007011	-0.000869	-0.000547	0.015823	-0.001600	-0.000190	0.006142	0.002023	-0.0
24	0.007163	-0.000461	-0.000325	0.015723	-0.002071	-0.001254	0.006522	0.003335	-0.0
25	0.007361	0.000134	-0.000107	0.015611	-0.002441	-0.001336	0.007191	0.004879	-0.0
26	0.007309	0.000308	0.000027	0.015988	-0.003080	-0.000784	0.007798	0.006687	-0.0
27	0.007302	0.000119	-0.000009	0.015947	-0.003175	-0.000457	0.007640	0.008156	-0.0
28	0.007545	0.000099	-0.000372	0.015488	-0.002948	-0.000009	0.007074	0.009315	-0.0
29	0.007711	0.000230	-0.000800	0.015021	-0.002766	0.000189	0.006419	0.010000	-0.0
...	...	...	...	...	...	...	...	...	
39970	0.006388	-0.004759	0.005731	-0.007059	-0.003124	0.008959	0.008993	0.004927	0.0
39971	0.006241	-0.005032	0.005508	-0.006725	-0.003911	0.008403	0.009166	0.005062	0.0
39972	0.005973	-0.005481	0.004933	-0.006659	-0.004468	0.008327	0.009479	0.005155	0.0
39973	0.006091	-0.005742	0.004307	-0.006997	-0.004808	0.009408	0.009711	0.005404	0.0

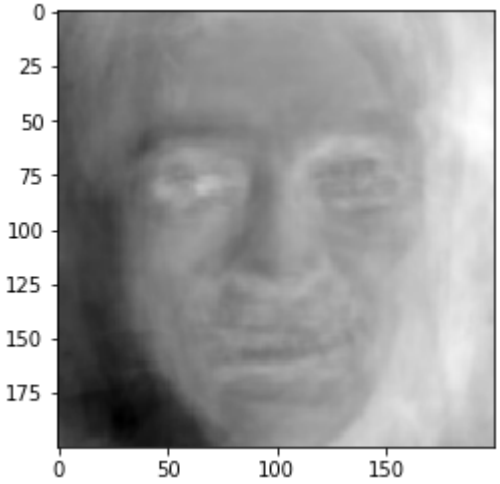
	0	1	2	3	4	5	6	7	
39974	0.006204	-0.005952	0.003663	-0.007282	-0.005169	0.010629	0.009826	0.005628	0.0
39975	0.006422	-0.005949	0.003307	-0.007710	-0.004232	0.012285	0.009977	0.005177	0.0
39976	0.006607	-0.005918	0.003136	-0.007935	-0.003452	0.013241	0.010378	0.004992	0.0
39977	0.006547	-0.006005	0.002919	-0.007906	-0.003486	0.012927	0.010308	0.006025	0.0
39978	0.006539	-0.006221	0.002835	-0.008109	-0.003622	0.012771	0.010250	0.006743	0.0
39979	0.006553	-0.006533	0.002766	-0.008090	-0.003967	0.012650	0.009888	0.006956	0.0
39980	0.006585	-0.006926	0.002753	-0.008156	-0.004343	0.012257	0.009388	0.006366	0.0
39981	0.006620	-0.007404	0.003152	-0.008543	-0.004868	0.011826	0.009821	0.005210	0.0
39982	0.006850	-0.007478	0.003266	-0.008844	-0.004975	0.012011	0.010501	0.004099	0.0
39983	0.006849	-0.007575	0.003174	-0.009188	-0.004895	0.011802	0.011195	0.004055	0.0
39984	0.006644	-0.007648	0.003324	-0.009509	-0.005188	0.011006	0.011216	0.004119	0.0
39985	0.006666	-0.007495	0.003390	-0.009880	-0.005456	0.010157	0.011430	0.004532	0.0
39986	0.006884	-0.007143	0.003419	-0.010148	-0.005379	0.009373	0.011875	0.004989	0.0
39987	0.006417	-0.007651	0.003542	-0.010217	-0.005708	0.008452	0.011525	0.006236	0.0
39988	0.005877	-0.008182	0.003518	-0.009754	-0.006361	0.007562	0.010981	0.006891	0.0
39989	0.005574	-0.008617	0.003706	-0.009806	-0.007334	0.006410	0.010101	0.007846	0.0
39990	0.005519	-0.008712	0.003689	-0.009765	-0.008213	0.005437	0.009495	0.008013	0.0
39991	0.005512	-0.008692	0.003378	-0.009982	-0.007875	0.005026	0.009987	0.008806	0.0
39992	0.005700	-0.008441	0.003184	-0.010300	-0.007523	0.005168	0.010498	0.008333	0.0
39993	0.005864	-0.008195	0.003069	-0.010225	-0.007338	0.005575	0.010791	0.007557	0.0
39994	0.005888	-0.008074	0.003166	-0.009841	-0.007027	0.005499	0.010915	0.007072	0.0
39995	0.005817	-0.008184	0.003298	-0.009345	-0.006289	0.005481	0.010674	0.006232	0.0
39996	0.005745	-0.008389	0.003297	-0.008926	-0.005108	0.005388	0.010107	0.005158	0.0
39997	0.005672	-0.008623	0.003353	-0.008578	-0.004149	0.005533	0.009815	0.004399	0.0
39998	0.005594	-0.008916	0.003201	-0.008236	-0.003235	0.006115	0.009697	0.003670	0.0
39999	0.005530	-0.009074	0.003047	-0.008118	-0.002671	0.006434	0.009648	0.003460	0.0

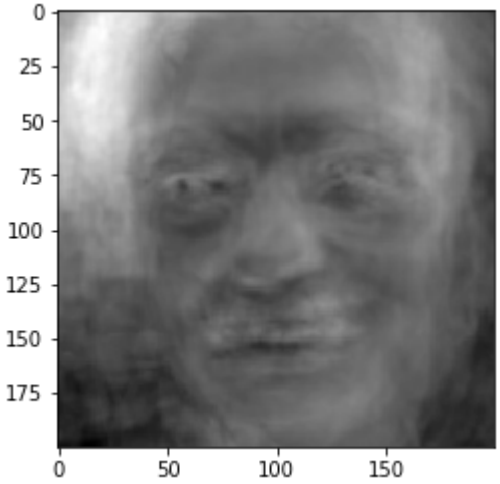
40000 rows × 20 columns

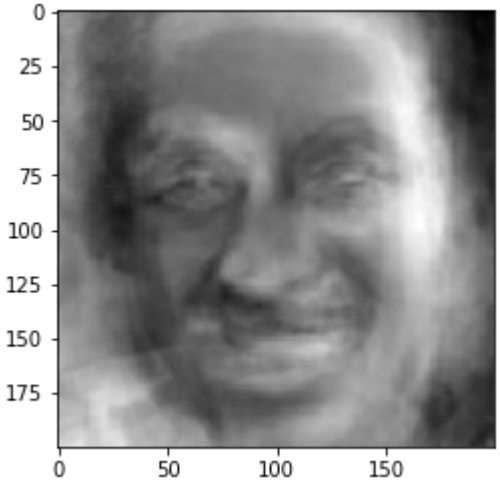
In [8]:
```
for colmn in range(0,ncompnts):
    z1=principalDf[colmn].values.reshape((200,200))
    plt.imshow(z1,cmap=plt.get_cmap('gray'))
    plt.show()
```

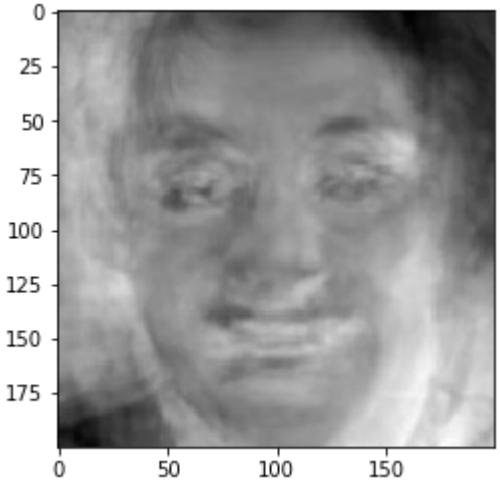

0
25
50
75
100
125
150
175
0
50
100
150

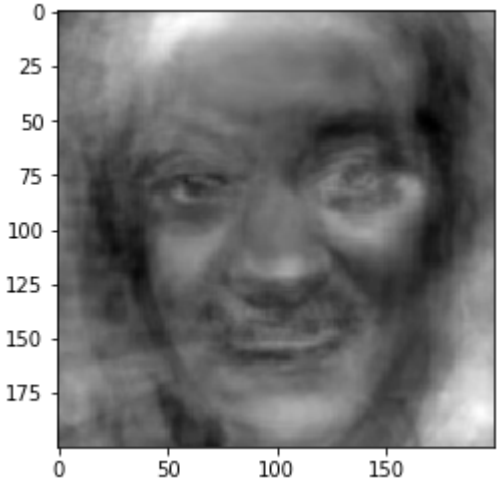

0
25
50
75
100
125
150
175
0
50
100
150

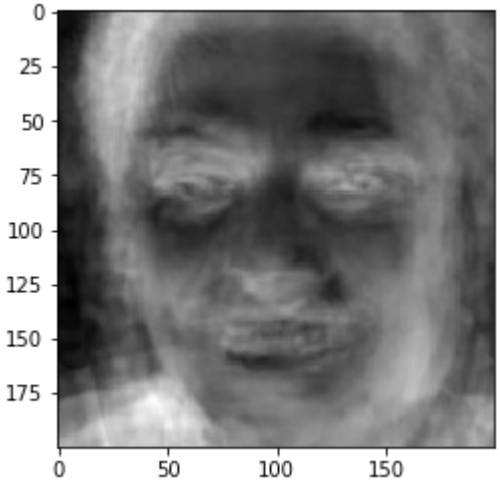

0
25
50
75
100
125
150
175
0
50
100
150

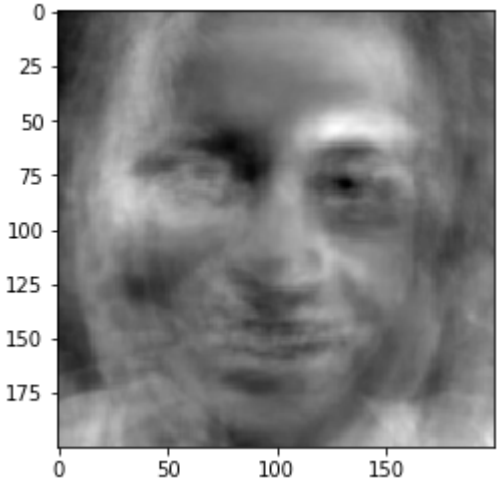

0
25
50
75
100
125
150
175
0
50
100
150

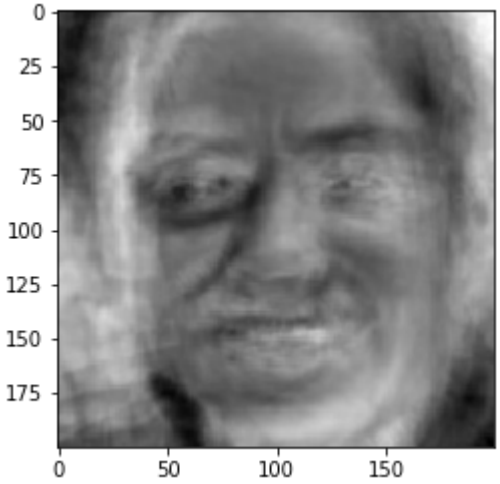

0
25
50
75
100
125
150
175
0
50
100
150

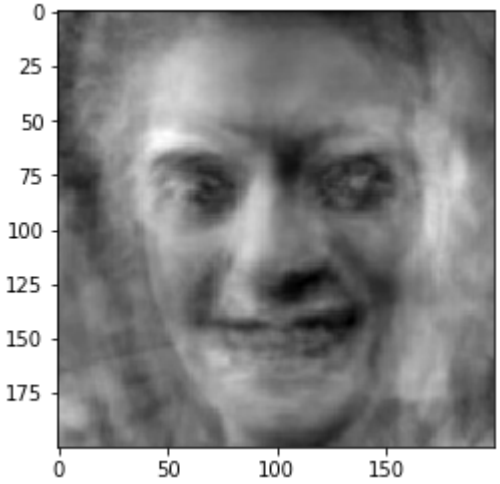

0
25
50
75
100
125
150
175
0
50
100
150

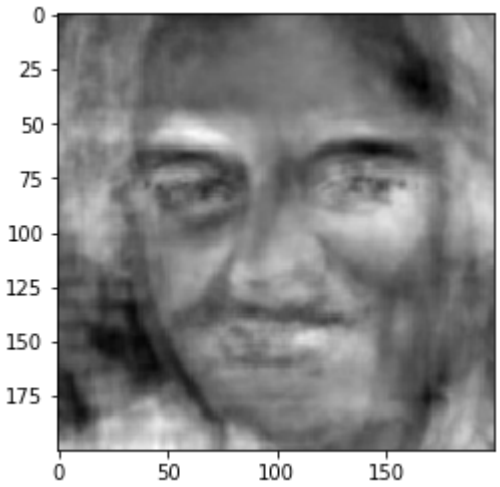

0
25
50
75
100
125
150
175
0
50
100
150

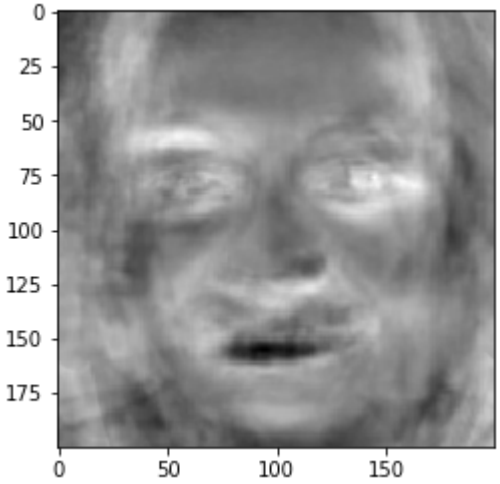

0
25
50
75
100
125
150
175
0
50
100
150

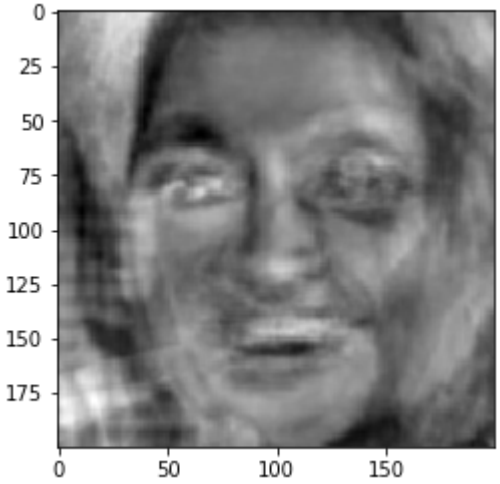

0
25
50
75
100
125
150
175
0
50
100
150

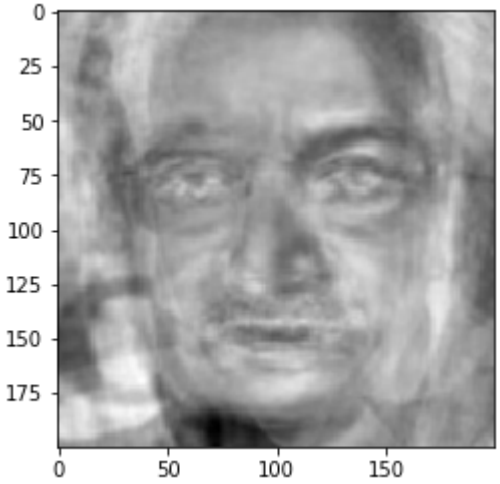

0
25
50
75
100
125
150
175
0
50
100
150

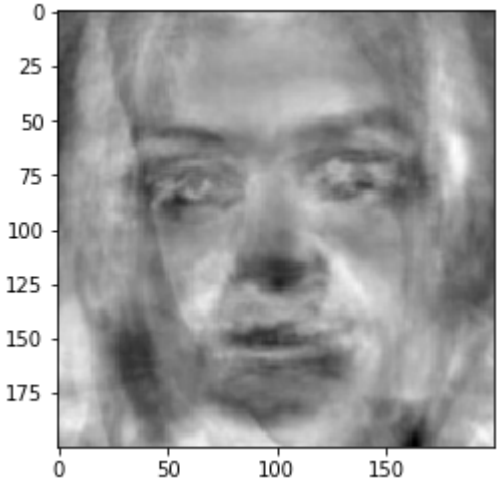

0
25
50
75
100
125
150
175
0
50
100
150

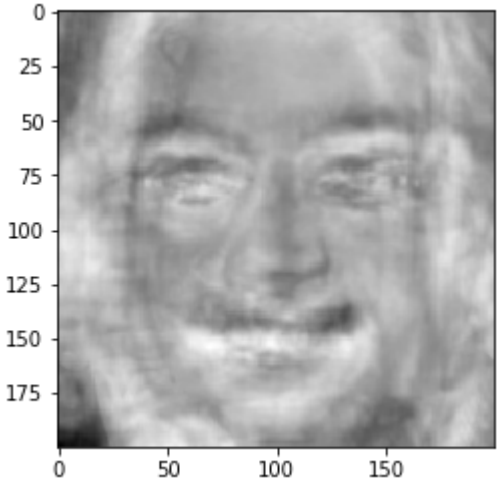

0
25
50
75
100
125
150
175
0
50
100
150

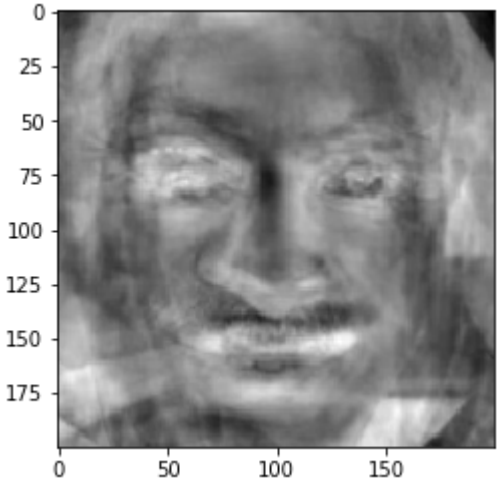

0
25
50
75
100
125
150
175
0
50
100
150

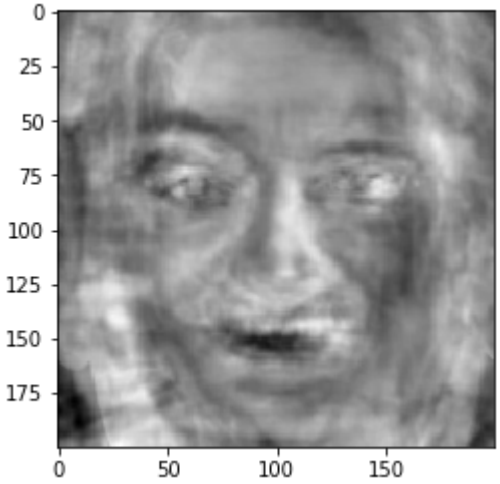

0
25
50
75
100
125
150
175
0
50
100
150

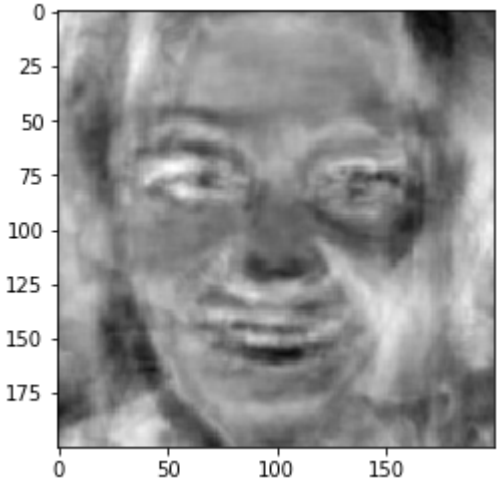

0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150


0
25
50
75
100
125
150
175
0
50
100
150





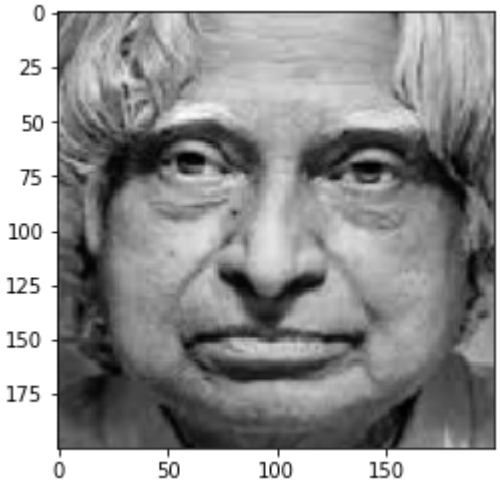
In [20]:
```
#KNN classification

knn_classifier = KNeighborsClassifier(n_neighbors = 1)
knn_classifier.fit(NMtransform, img_class.T.ravel())
```

Out[20]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=1, p=2,
                     weights='uniform')
```

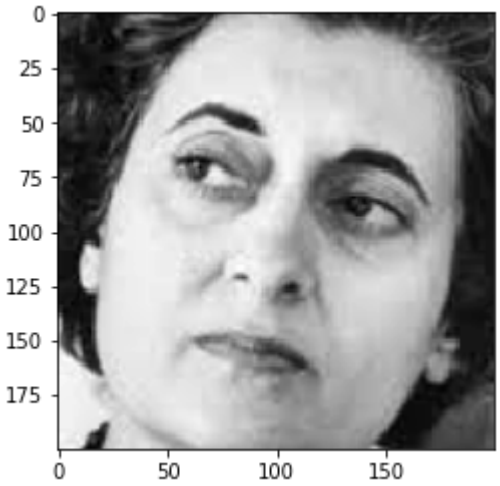
In [21]:
```
print('Image Testing')

str="G:/Yamuna docs/College docs/Info Ret/Extra credit/test_img/" #test image
 database set

entries=os.listdir(str)
set=1
label_mng= pd.DataFrame(data=[['apj','Abdul Kalam'],['ind', 'Indira Gandhi'],[
'kmj','Kamaraj'],['pv','PV Sindhu'],['rd','Rahul Dravid']
,['sc','Sachin'],['sn','Sreenath'],['tc','Saina Nehwal'],['p','Karthik'],['kc'
,'Kalpana Chawla'],['az','Azim premji']],columns= ['Label',

'Name'])
print(label_mng)
for entry in entries:
    if entry.endswith(".jpg") or entry.endswith(".jfif"):
#         print(entry)
        image = cv2.imread(str + entry)
#         plt.imshow(image)
#         plt.show()

        img_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
#         plt.imshow(img_rgb)
#         plt.show()

        img_gray=cv2.cvtColor(img_rgb,cv2.COLOR_BGR2GRAY)
#         plt.imshow(img_gray,cmap=plt.get_cmap('gray'))
#         plt.show()


        faces = face_cascade.detectMultiScale(img_gray, 1.3, 5)
        if len(faces)>0:
#         print(faces)
            for (x,y,w,h) in faces:
                roi_gray = img_gray[y:y+h, x:x+w]



            roi_gray_resize=cv2.resize(roi_gray,(200,200))
            plt.imshow(roi_gray_resize,cmap=plt.get_cmap('gray'))
            plt.show()

#         cv2.imshow('img',roi_gray_resize)
#         cv2.waitKey(0)
#         cv2.destroyAllWindows()

            fa=np.array(roi_gray_resize)
            fa=fa.flatten()
            m=entry.find('.')
            fname=entry[0:m]

    # Standardizing the Observations
            Te=pd.DataFrame(fa)
            TM=Te.sub(Mmean,axis='rows')
            TMPC= pca.transform(TM.T)
            y_pred=knn_classifier.predict(TMPC)
```

```
            print('Predicted-->',y_pred)

#         Appro=pca.inverse_transform(TMPC)
#         Approdf=pd.DataFrame(Appro)
#         Approdf=Approdf.add(Mmean,axis='rows')
#     for n in range(0,21):
#         fig = plt.figure()
#         a = fig.add_subplot(1, 2, 2)
#         z1=np.reshape(Appro[:,n],(200,200))
#         plt.imshow(z1,cmap=plt.get_cmap('gray'))
#         a.set_title('After')
#         a = fig.add_subplot(1, 2, 1)
#         z1=np.reshape(fa,(200,200))
#         plt.imshow(z1,cmap=plt.get_cmap('gray'))
#         a.set_title('Before')
##

#
```

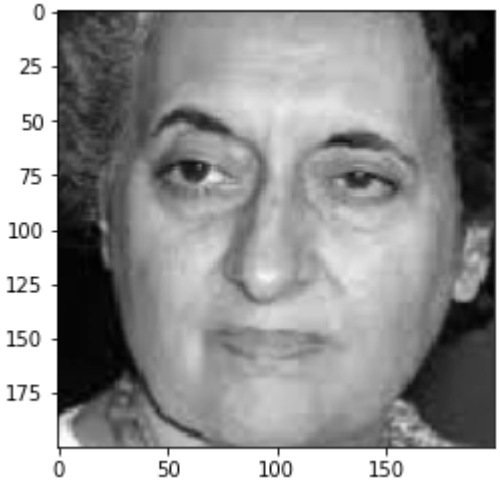
```
Image Testing
   Label            Name
0    apj     Abdul Kalam
1    ind   Indira Gandhi
2    kmj         Kamaraj
3     pv       PV Sindhu
4     rd    Rahul Dravid
5     sc          Sachin
6     sn        Sreenath
7     tc    Saina Nehwal
8      p         Karthik
9     kc  Kalpana Chawla
10    az     Azim premji
```



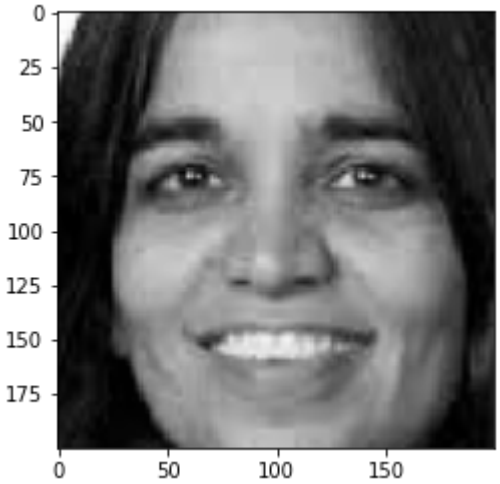

```
Predicted--> ['apj']
```



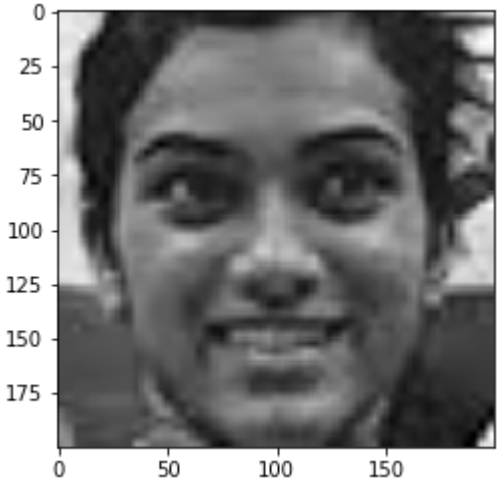

```
Predicted--> ['ind']
```



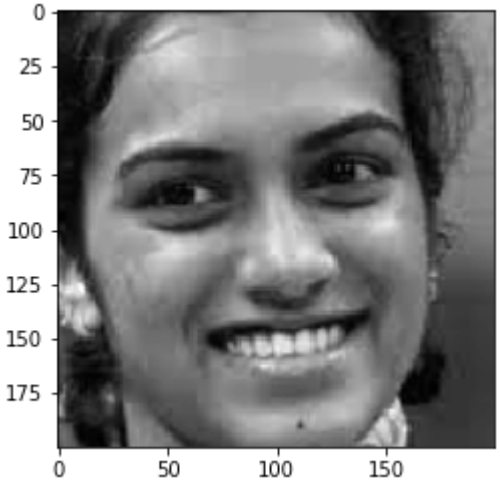

```
Predicted--> ['rd']
```



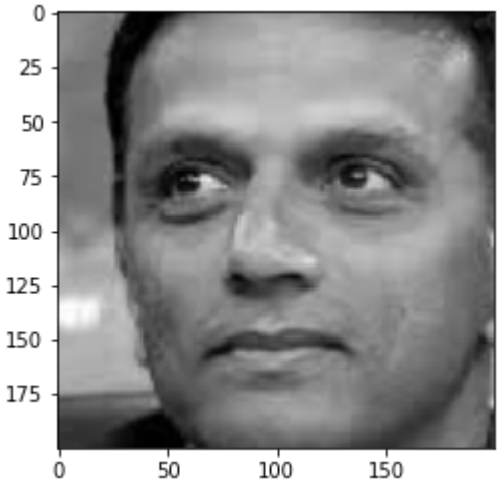

```
Predicted--> ['kc']
```



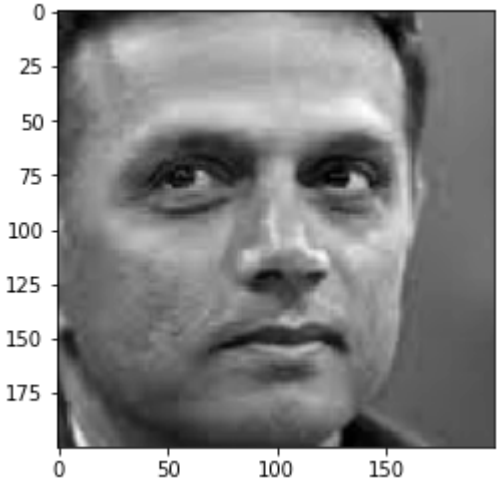

```
Predicted--> ['sc']
```



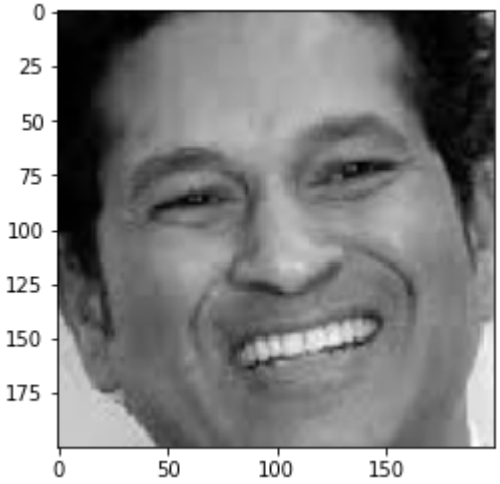

```
Predicted--> ['pv']
```



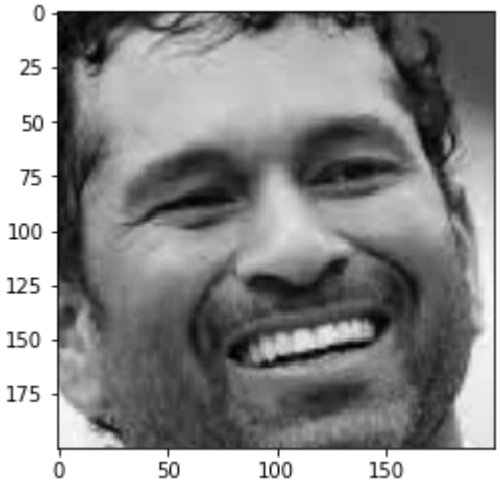

```
Predicted--> ['sn']
```



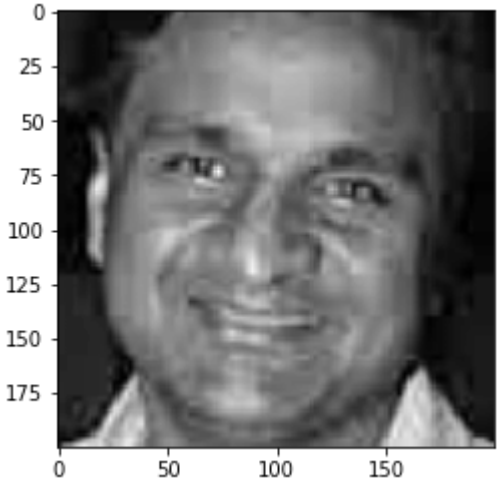

```
Predicted--> ['pv']
```



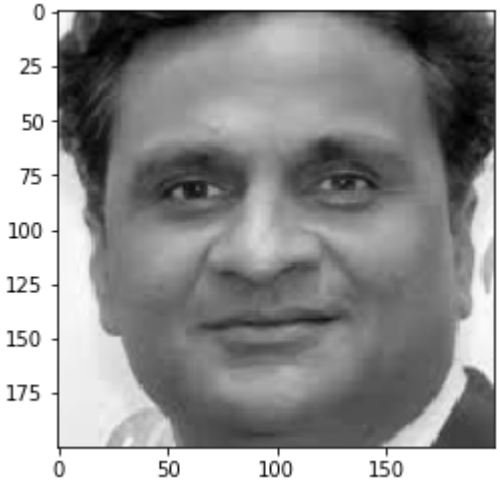

```
Predicted--> ['sc']
```



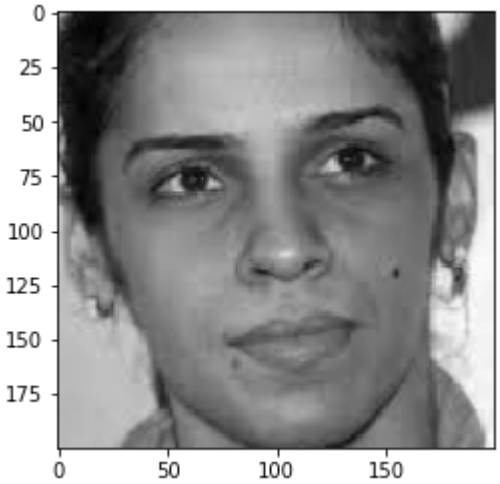

```
Predicted--> ['tc']
```



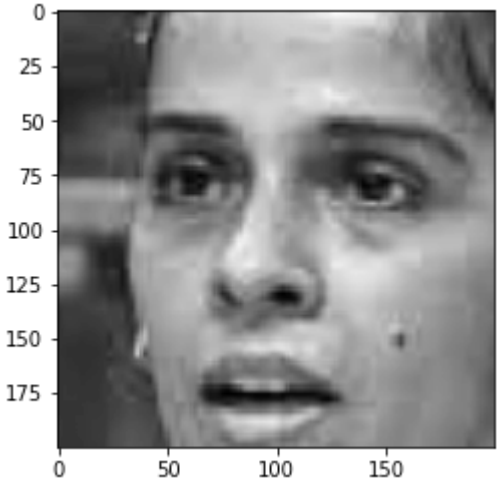

```
Predicted--> ['sn']
```




```
Predicted--> ['sc']
```




```
Predicted--> ['sc']
```




```
Predicted--> ['pv']
```

In [ ]: