Biometric solution for 2D faces captured in non-controlled environments

Realised by:
Nada BELAIDI
Khaled CHARAABI
Racha MAGHERBI
Tutored by:
Nefissa Khiari

TABLE OF CONTENTS

01
INTRODUCTION

02
BUSINESS UNDERSTANDING

03
DATA SCIENCE OBJECTIVES

04
DATA COMPREHENSION

05
DATA MODELING AND EVALUATION

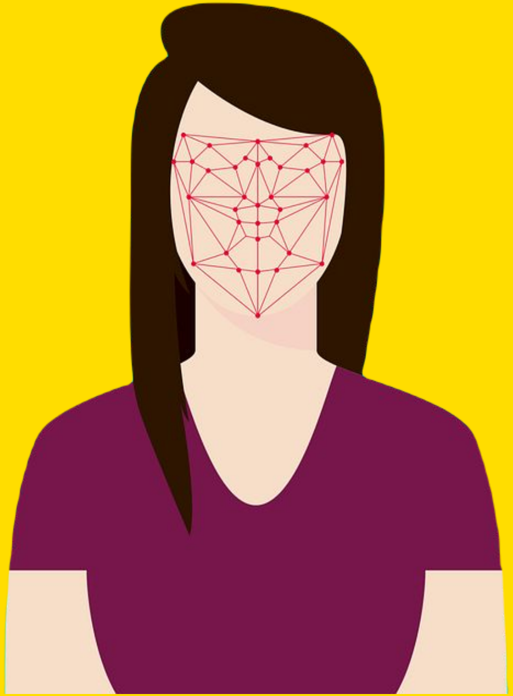
01

INTRODUCTION

INTRODUCTION

Facial recognition is a biometric technology that uses distinguishable facial features to identify a person.

Today, it's used in a variety of ways from allowing you to unlock your phone, go through security at the airport , purchase products at stores ect…



02

BUSINESS UNDERSTANDING

Business understanding

Face recognition :

-Helps find missing people and identify perpetrators

-Protects businesses against theft

-Strengthens security measures in banks and airports

-Makes shopping more efficient

-Reduces the number of touchpoints

-Improves photo organization

-Improves medical treatment

-Saves time

03

DATA OBJECTIVES

Data objectives

Feature engineering	Data understanding
Data preparation	Preparation of inter and intra class comparison lists.
Data modeling	Classification Scoring

-Data Science Objectives :

- Developing a solution for automatic recognition of 2D captured faces in uncontrolled environments

  → maintain a high recognition rate despite the conditions

  → Recognizing correctly detected faces

  → Modernize the environment

- Identifying the suitable technologies for our business objectives.

- Training fast and efficient Deep Learning models.

  Key Results :

  - Using MTCNN for face detection..
  - Using Cascade classifier for face detection..
  - Using SVM for face recognition.
  - Using Facenet for face recognition.

04

DATA COMPREHENSION

Face Recognition Dataset, a database of face photographs designed for the creation of face detection and recognition models. This dataset has been derived from the Labeled Faces in the Wild Dataset.

This dataset is a collection of JPEG pictures of famous people collected on the internet.

The Dataset : Each picture is centered on a single face, and every image is encoded in RGB. The original images are of the size 250 x 250. The dataset contains 1680 directories, each representing a celebrity. Each directory has 2-50 images for the celebrity.

Data Preprocessing

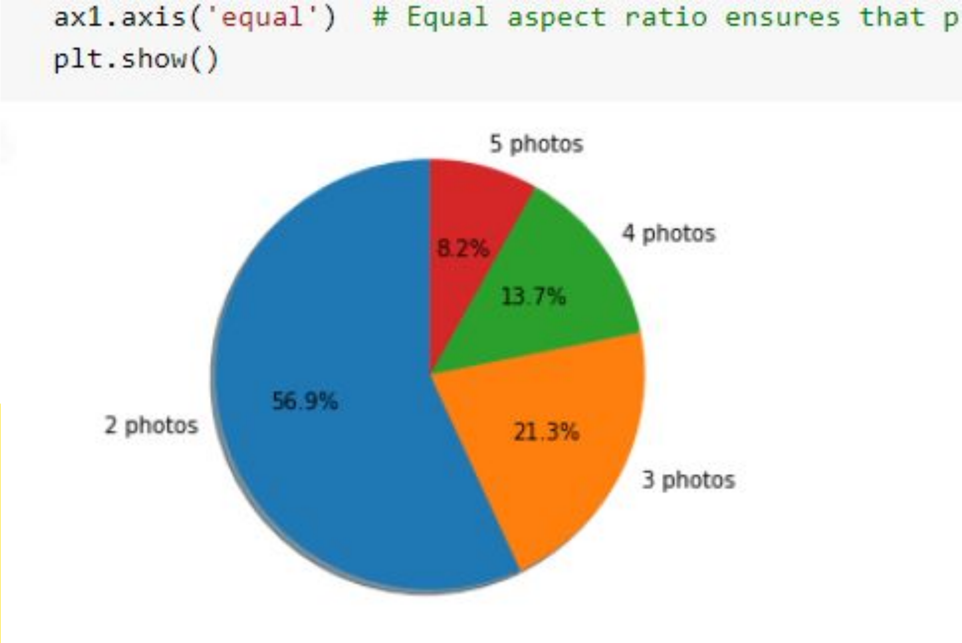
we notice the existence of 143 people who have more than 10 photos going up to 530 photos which will lead to the overfitting of the models from where we decide to eliminate them.
we will stick to :

categorie 2 photos : 779 personnes

categorie 3 photos : 291 personnes

categorie 4 photos : 187 personnes

categorie 5 photos : 112 personnes

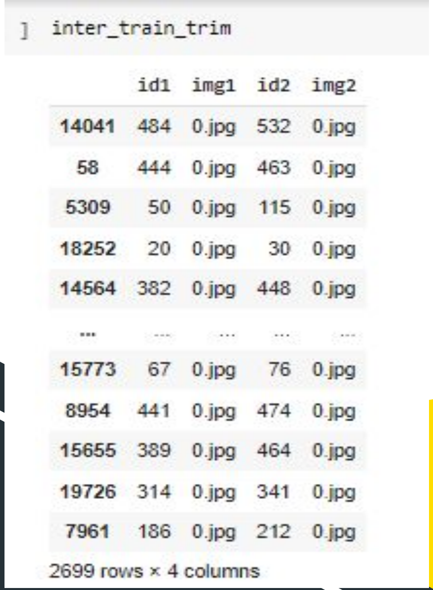


```
ax1.axis('equal')  # Equal aspect ratio ensures that p
plt.show()
```

Preparation of inter and intra class comparison lists

We organize the dataset in the form of 70% of data of each category in trainset and 30% of each category in testset.

These results are all the possible combinations for intra-class comparisons, but we must have the same number of intra-class comparisons as between classes. Thus, we extract randomly from intra_train and intra_test to have the same number of lines as inter_train and inter_test respectively.

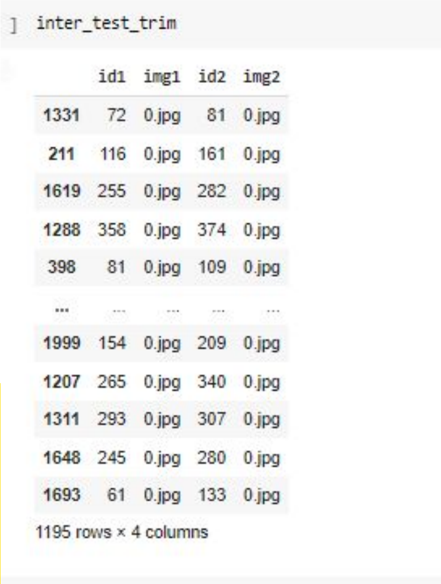


```
] inter_train_trim
```

	id1	img1	id2	img2
14041	484	0.jpg	532	0.jpg
58	444	0.jpg	463	0.jpg
5309	50	0.jpg	115	0.jpg
18252	20	0.jpg	30	0.jpg
14564	382	0.jpg	448	0.jpg
...	...	...	...	...
15773	67	0.jpg	76	0.jpg
8954	441	0.jpg	474	0.jpg
15655	389	0.jpg	464	0.jpg
19726	314	0.jpg	341	0.jpg
7961	186	0.jpg	212	0.jpg

2699 rows × 4 columns

inter-class comparaison lists(train)

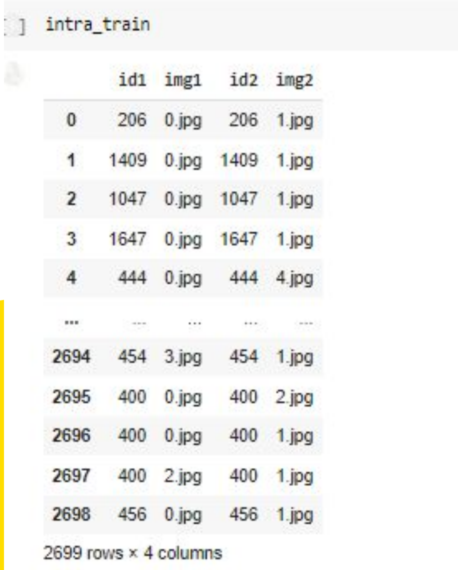


```
] inter_test_trim
```

	id1	img1	id2	img2
1331	72	0.jpg	81	0.jpg
211	116	0.jpg	161	0.jpg
1619	255	0.jpg	282	0.jpg
1288	358	0.jpg	374	0.jpg
398	81	0.jpg	109	0.jpg
...	...	...	...	...
1999	154	0.jpg	209	0.jpg
1207	265	0.jpg	340	0.jpg
1311	293	0.jpg	307	0.jpg
1648	245	0.jpg	280	0.jpg
1693	61	0.jpg	133	0.jpg

1195 rows × 4 columns

-inter-class comparaison lists(test)

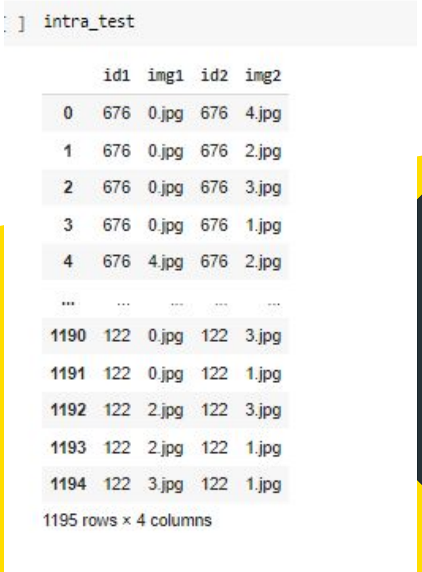


```
] intra_train
```

	id1	img1	id2	img2
0	206	0.jpg	206	1.jpg
1	1409	0.jpg	1409	1.jpg
2	1047	0.jpg	1047	1.jpg
3	1647	0.jpg	1647	1.jpg
4	444	0.jpg	444	4.jpg
...	...	...	...	...
2694	454	3.jpg	454	1.jpg
2695	400	0.jpg	400	2.jpg
2696	400	0.jpg	400	1.jpg
2697	400	2.jpg	400	1.jpg
2698	456	0.jpg	456	1.jpg

2699 rows × 4 columns

-intra-class comparaison lists(train)



```
] intra_test
```

	id1	img1	id2	img2
0	676	0.jpg	676	4.jpg
1	676	0.jpg	676	2.jpg
2	676	0.jpg	676	3.jpg
3	676	0.jpg	676	1.jpg
4	676	4.jpg	676	2.jpg
...	...	...	...	...
1190	122	0.jpg	122	3.jpg
1191	122	0.jpg	122	1.jpg
1192	122	2.jpg	122	3.jpg
1193	122	2.jpg	122	1.jpg
1194	122	3.jpg	122	1.jpg

1195 rows × 4 columns

intra-class comparaison lists(test)

05

Data Modeling and Evaluation

Modeling is the phase in which our work begins to be more clear. After and building all the necessary data, the next step in this project is to create the most suitable model that will meet our business objectives.

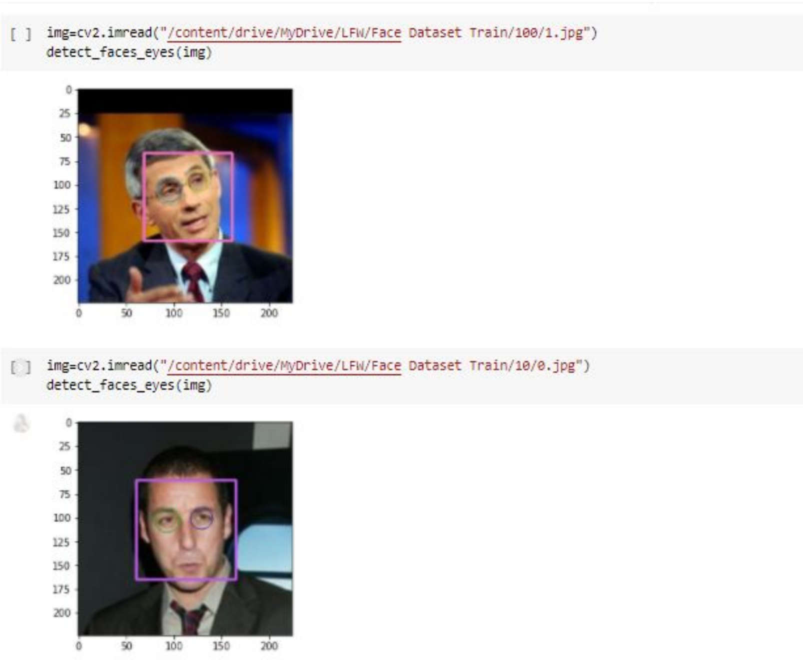
In this section, the development of different techniques used will be discussed and explained thorough

I-Face Detection

For the face detection we ve chosen 2 models : Cascade Classifier and MTCNN

1- Cascade Classifier:

It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

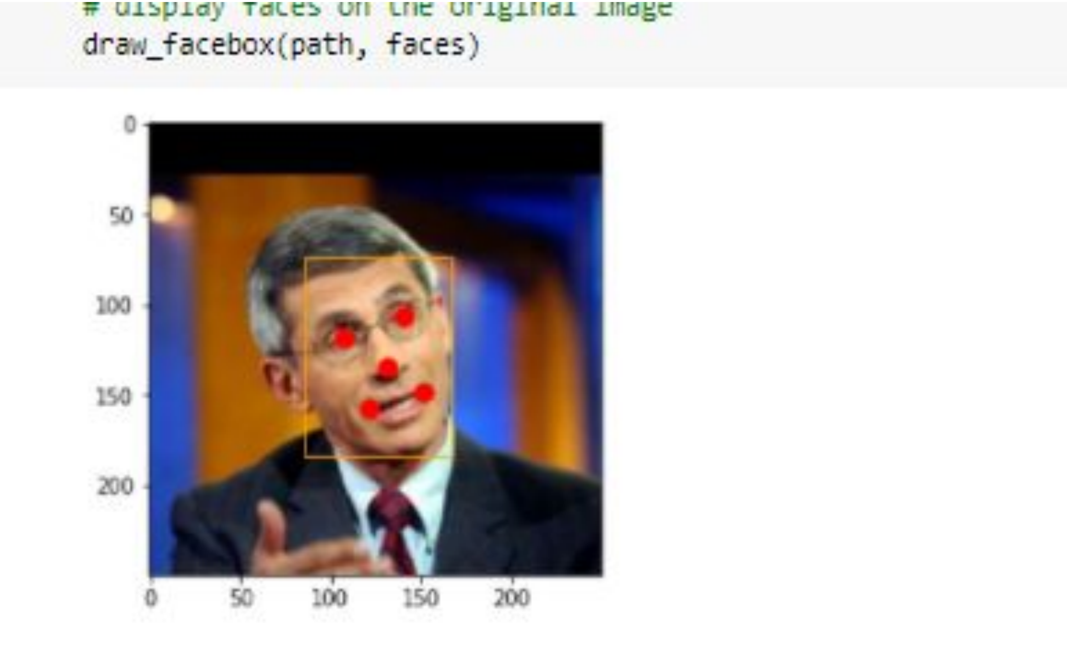


```
img=cv2.imread("/content/drive/MyDrive/LFW/Face Dataset Train/100/1.jpg")
detect_faces_eyes(img)
```

```
img=cv2.imread("/content/drive/MyDrive/LFW/Face Dataset Train/10/0.jpg")
detect_faces_eyes(img)
```

2-MTCNN :

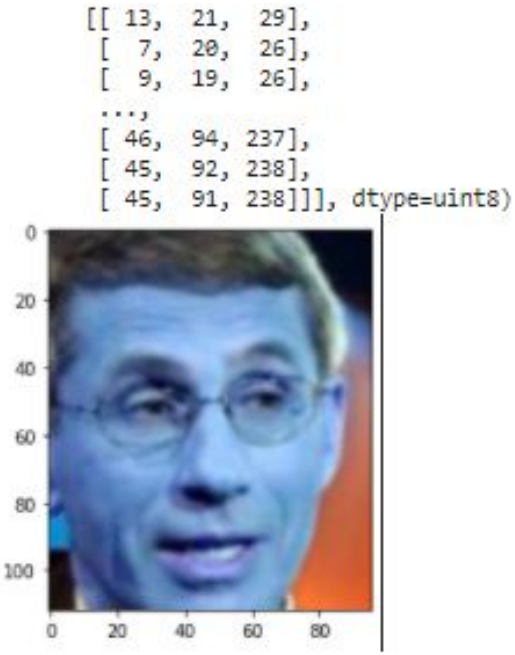
Multi-Task Cascaded Convolutional Neural Networks is a neural network which detects faces and facial landmarks on images.



```
# display faces on the original image
draw_facebox(path, faces)
```

II-Face alignment

The results we got from MTCNN were more accurate so we used them for face alignment.

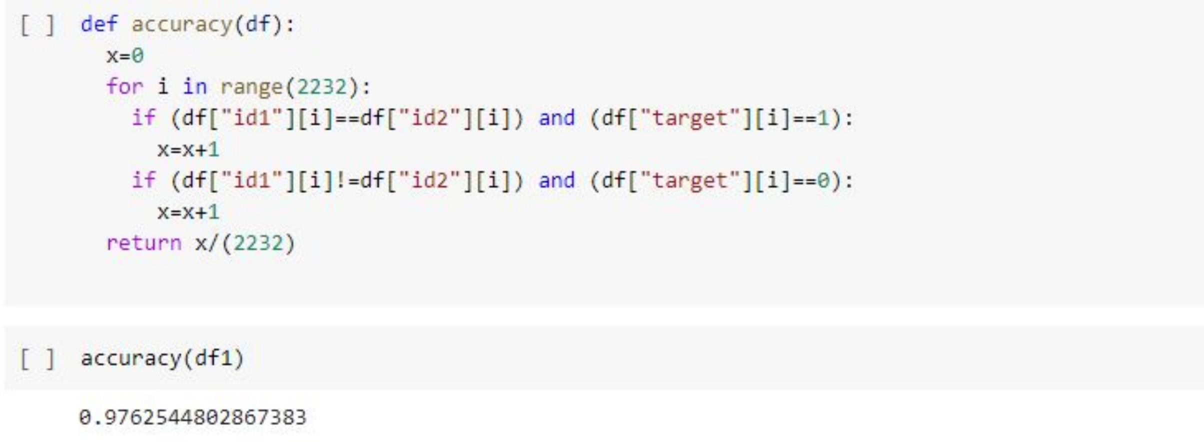


```
[[ 13,  21,  29],
 [  7,  20,  26],
 [  9,  19,  26],
 ...,
 [ 46,  94, 237],
 [ 45,  92, 238],
 [ 45,  91, 238]]], dtype=uint8)
```

III-Face Recognition

After finishing the process of face detection , *recognition* is the attempt to confirm the identity of persons which is use for verification . We used two different models Facenet and SVM.

1-Facenet:
A one-shot model that directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity.
After training our model with our dataset our testing results were 97,6% accurate.



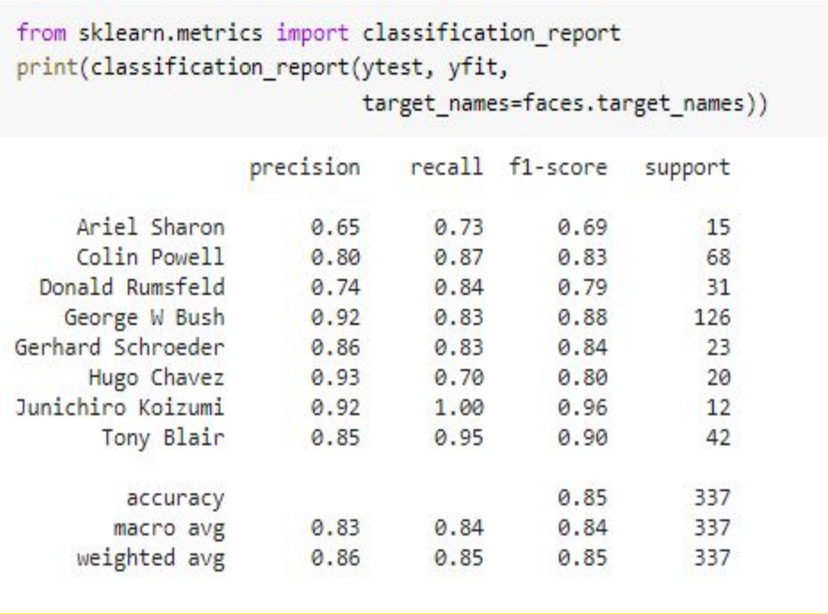
```
[ ]  def accuracy(df):
       x=0
       for i in range(2232):
         if (df["id1"][i]==df["id2"][i]) and (df["target"][i]==1):
           x=x+1
         if (df["id1"][i]!=df["id2"][i]) and (df["target"][i]==0):
           x=x+1
       return x/(2232)
```

```
[ ]  accuracy(df1)

     0.9762544802867383
```

2-SVM :

algorithm generates a decision surface separating the two classes. For face recognition, we re-interpret the decision surface to produce a similarity metric between two facial images. These were our testing results on our dataset.

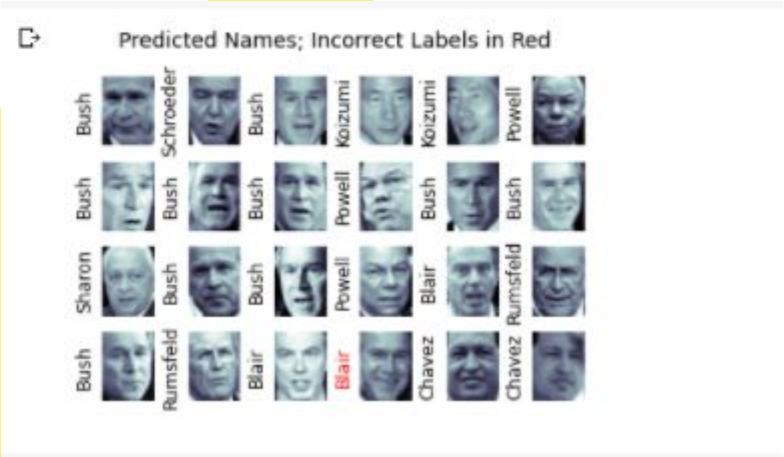


```python
from sklearn.metrics import classification_report
print(classification_report(ytest, yfit,
                            target_names=faces.target_names))
```

```
                   precision    recall  f1-score   support

     Ariel Sharon       0.65      0.73      0.69        15
     Colin Powell       0.80      0.87      0.83        68
  Donald Rumsfeld       0.74      0.84      0.79        31
    George W Bush       0.92      0.83      0.88       126
Gerhard Schroeder       0.86      0.83      0.84        23
      Hugo Chavez       0.93      0.70      0.80        20
Junichiro Koizumi       0.92      1.00      0.96        12
       Tony Blair       0.85      0.95      0.90        42

         accuracy                           0.85       337
        macro avg       0.83      0.84      0.84       337
     weighted avg       0.86      0.85      0.85       337
```



Predicted Names; Incorrect Labels in Red

Thank you for your attention