# Face Classification

## Machine Learning project on Olivetti Dataset

**PwC Tech Wednesday 20/Mar/2024**

Author Ing. Vittorio Stile

Universitas Mercatorum

Università telematica delle Camere di Commercio Italiane

**DOTTORATO INDUSTRIALE IN BIG DATA ED INTELLIGENZA ARTIFICIALE (XXXVIII CICLO)**

*Curricula "Big data management per la transizione digitale" Università delle Camere di Commercio Italiane "UNIVERSITAS MERCATORUM"*

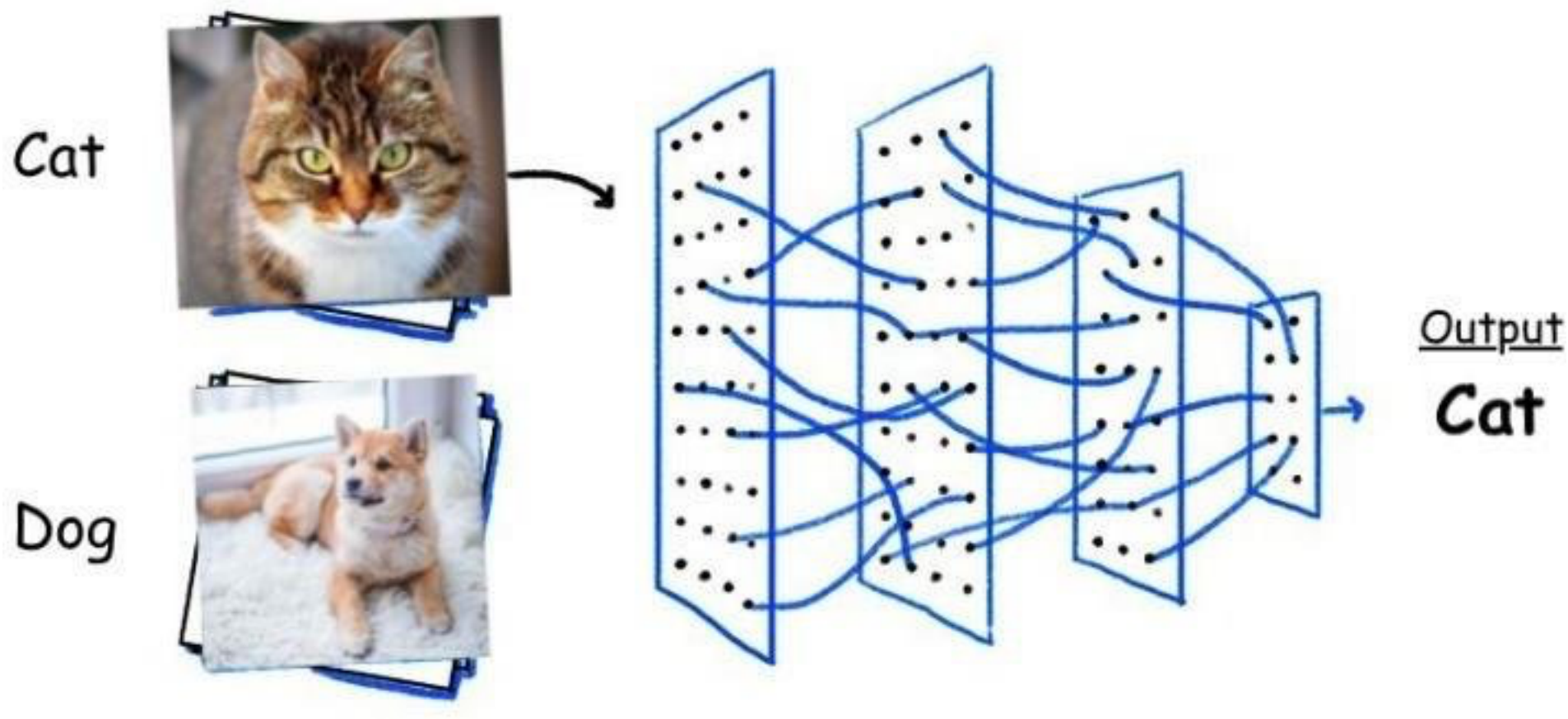
**DOTTORANDO:** VITTORIO STILE
**TUROR ACCADEMICO:** PROF. ROBERTO CALDELLI
**TUROR AZIENDALE:** DOTT.SSA ELENA SANTI
**IMPRESA:** PRICEWATERHOUSECOOPERS BUSINESS SERVICES SRL

# INTRODUCTION

In our technology-driven world, classification and face recognition play crucial roles in enhancing various aspects of our daily lives (privacy, security, accessibility).

These technologies leverage the power of artificial intelligence and machine learning to organise information efficiently and enable advanced applications.

# WHAT IS CLASSIFICATION?



# WHAT IS CLASSIFICATION? II

Classification is a fundamental concept in machine learning where an algorithm is trained to categorise input data into distinct classes or groups. The goal is to learn a mapping from input features to predefined output labels, enabling the algorithm to make predictions on new, unseen data.

- **Categorisation**: The process involves assigning input data points to predefined categories or classes.
- **Training Data:** Algorithms learn from a labeled dataset, where examples have known outcomes.
- **Predictive Capability:** Once trained, the algorithm can generalize its learning to classify new, previously unseen data. For example, recognising different types of animals that have been fed to the algorithm as training data.

# IMPORTANCE OF CLASSIFICATION:

Classification is fundamental to organising and making sense of vast amounts of data. In a world flooded with information, classification systems help us categorise and understand the world around us. This is vital for:

1. **Search Engines**: Search engines like Google use classification algorithms to provide relevant search results quickly.

2. **E-commerce:** Online retailers employ classification to recommend products based on customers' preferences and purchase history.

3. **Healthcare**: Classification assists in medical diagnosis by categorising symptoms and predicting potential diseases.

# TYPES OF CLASSIFICATION

Face recognition is a subset of classification that focuses on identifying and verifying individuals based on facial features. Its significance is evident in several real-world applications:

1. **Supervised Learning:** algorithm is trained on a labeled dataset, ex. Hand written digits, labeled faces
2. **Unsupervised Learning:** algorithms that identify patterns and relationships within data without labeled ex. anomaly detection
3. **Semi-supervised Learning:** leverages a limited amount of labeled data along with a larger pool of unlabelled data, ex. Voice recognition

# TYPES OF CLASSIFICATION

| Items | Supervised learning | Semi-supervised learning | Unsupervised learning |
|---|---|---|---|
| Input type | Labelled data | A mixture of labelled and unlabelled data | Unlabelled data |
| Accuracy | High | Mid | Low |
| Complexity of the algorithm | Low | Mid | High |
| Types of algorithm | Regression and classification | Regression, classification, clustering, and association | Clustering and association |

# FACE CLASSIFICATION

Face recognition is a subset of classification that focuses on identifying and verifying individuals based on facial features. Its significance is evident in several real-world applications:

1. **Security**: Face recognition is widely used in security systems, such as unlocking smartphones, access control in buildings, and surveillance cameras.
2. **Social Media:** Platforms like Facebook use face recognition for photo tagging, making it easier for users to identify and share images of friends and family.
3. **Authentication**: Biometric authentication, including facial recognition, is employed for secure access to devices, apps, and sensitive information.
4. **Law Enforcement:** Face recognition assists law enforcement agencies in identifying and tracking individuals in public spaces, aiding in criminal investigations.

# HOW DOES FACE RECOGNITION NORMALLY WORK?!

1. The computer analyses the captured image to extract distinctive features, such as the size and shape of the eyes, nose, and mouth.
2. These extracted features are converted into a numerical / vector representation, creating a unique "faceprint" for that individual.
3. The computer compares the generated faceprint with a dataset or a database of known faceprints.
4. The system searches for a match between the newly captured faceprint and the stored faceprints in the database.
5. If a match is found with sufficient confidence, the system identifies the person. Otherwise, it may prompt for further verification.

# HOW DOES FACE RECOGNITION NORMALLY WORK?!

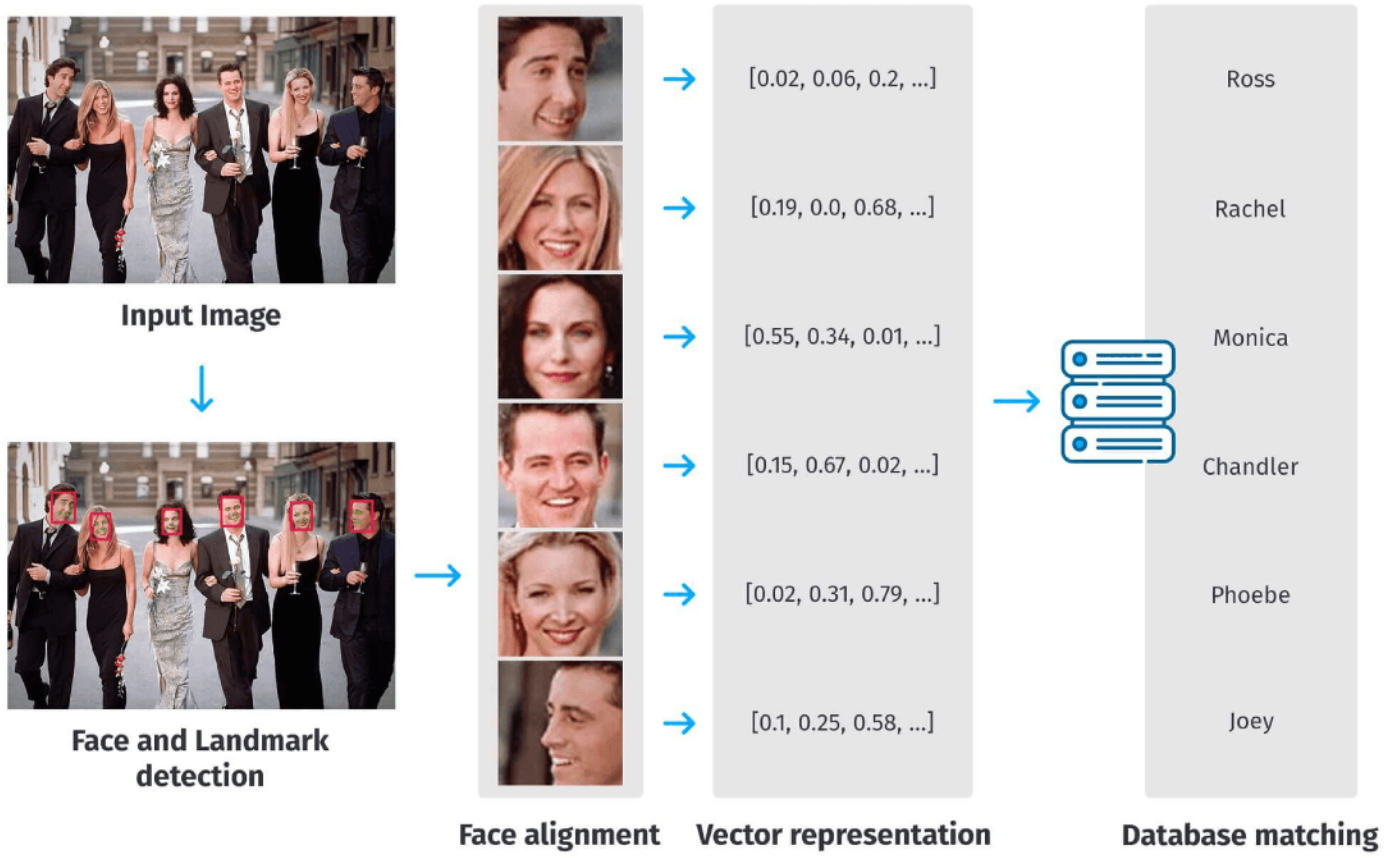



Figure 1. Schematic overview of the end-to-end face recognition system

# HOW DOES FACE RECOGNITION NORMALLY WORK?!

1. The computer analyses the captured image to extract distinctive features, such as the size and shape of the eyes, nose, and mouth.
2. These extracted features are converted into a numerical representation, creating a unique "faceprint" for that individual.
3. The computer compares the generated faceprint with a database of known faceprints. This database contains faceprints of individuals who have been previously enrolled in the system.
4. The system searches for a match between the newly captured faceprint and the stored faceprints in the database.
5. If a match is found with sufficient confidence, the system identifies the person. Otherwise, it may prompt for further verification.

# HANDS-ON

**WE ARE GOING TO DO A SIMPLE APPLICATION OF A FACE CLASSIFICATION ON THE OLIVETTI DATASET, BRIEF INFORMATION ABOUT OLIVETTI DATASET:**

- Face images taken between April 1992 and April 1994.
- There are ten different image of each of 40 distinct people
- There are 400 face images in the dataset
- Face images were taken at different times, variying ligthing, facial express and facial detail
- All face images have black background
- The images are gray level
- Size of each image is 64x64
- Image pixel values were scaled to [0, 1] interval
- Names of 40 people were encoded to an integer from 0 to 39

# THE PROJECT 1.01

## First we import necessary libraries

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import fetch_olivetti_faces
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score
```

import numpy as np: NumPy is a library for numerical operations in Python. We use it for efficient array manipulation.

import matplotlib.pyplot as plt: Matplotlib is a popular data visualization library. Here, it's used for displaying images.

from sklearn.datasets import fetch_olivetti_faces: Scikit-learn provides the Olivetti Faces dataset, which contains grayscale images of faces.

from sklearn.model_selection import train_test_split: This module helps in splitting the dataset into training and testing sets.

from sklearn.ensemble import RandomForestClassifier: Random Forest is an ensemble learning method used for classification.

from sklearn.metrics import classification_report, accuracy_score: Scikit-learn metrics for evaluating classification models.

# THE PROJECT 1.02

**Then we import the dataset and split it into training and target.**

```
data = fetch_olivetti_faces() #load dataset
X = data.images.reshape((len(data.images), -1)) #Training set
y = data.target #Target set
```

data = fetch_olivetti_faces(): Fetches the Olivetti Faces dataset

X = data.images.reshape((len(data.images), -1)): Reshapes the images into a flat array for model training.

y = data.target: Represents the target labels for each image.

# THE PROJECT 1.03

## Then we visualise the dataset.

```
fig, ax = plt.subplots(5, 8, figsize=(15, 10)) #Using matplot lib to visualize the images
ax = ax.flatten()

for i in range(40): #loop to show all 40 people
    idx = np.where(y == i)[0][0] #
    ax[i].imshow(X[idx].reshape(64, 64), cmap='gray')
    ax[i].set_title(f'Person {i}')
    ax[i].axis('off')

# Adjust layout
plt.tight_layout()
plt.show()
```

Utilizes Matplotlib with a loop to display a subset of the Olivetti Faces dataset, showcasing images of 40 individuals. We have to ensure all the images have the same shape 64x64 and they have the same color “grayscale”. In order to ensure best accuracy, so if we feed the model a grayscale and a coloured photo it will ‘confuse’ the model.

# THE PROJECT 1.03: OUTPUT

# THE PROJECT 1.04

**Then we split the training data again into a training set and a test set.**

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
#splitting into test set for testing
```

X: Represents the input features, in this case, the flattened images of faces.

y: Corresponds to the target labels, indicating the identity of the individuals in the images.

test_size=0.25: Specifies that 25% of the data will be reserved for testing, while the remaining 75% will be used for training.

random_state=42: Sets a seed for the random number generator to ensure reproducibility. The same seed produces the same split every time the code is run.

The line creates four sets:

X_train: The training set of input features.

X_test: The testing set of input features.

y_train: The corresponding training set of target labels.

y_test: The corresponding testing set of target labels.

Training Set: Used to train the machine learning model. The model learns patterns and relationships from this data.

Testing Set: Used to evaluate the model's performance. It ensures that the model can generalize well to new, unseen data.

# THE PROJECT 1.05

## We set up our model.

```
clf = RandomForestClassifier(n_estimators=100, random_state=42) #using a
random forest model for the classification task
clf.fit(X_train, y_train)
```

clf = RandomForestClassifier(n_estimators=100, random_state=42):

Initializes a Random Forest classifier with 100 trees.

clf.fit(X_train, y_train):

Trains the classifier on the training data.

# THE PROJECT 1.06

## Model Evaluation

```
y_pred = clf.predict(X_test) #testing on our test set
print("Accuracy:", accuracy_score(y_test, y_pred)) #accuracy score
print(classification_report(y_test, y_pred)) #classification report
```

y_pred = clf.predict(X_test): Predicts labels for the test set

print("Accuracy:", accuracy_score(y_test, y_pred)): Prints the accuracy of the model.

print(classification_report(y_test, y_pred)): Prints a detailed classification report.

# THE PROJECT 1.07: THE RESULTS

Accuracy: 0.88

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 0.50 | 0.67 | 4 |
| 1 | 1.00 | 1.00 | 1.00 | 2 |
| 2 | 0.50 | 1.00 | 0.67 | 2 |
| 3 | 1.00 | 1.00 | 1.00 | 4 |
| 4 | 1.00 | 0.67 | 0.80 | 3 |
| 5 | 0.75 | 1.00 | 0.86 | 3 |
| 6 | 1.00 | 1.00 | 1.00 | 1 |
| 7 | 1.00 | 0.43 | 0.60 | 7 |
| 8 | 0.67 | 1.00 | 0.80 | 2 |
| 9 | 0.75 | 1.00 | 0.86 | 3 |
| 10 | 0.75 | 1.00 | 0.86 | 3 |
| 11 | 1.00 | 0.75 | 0.86 | 4 |
| 12 | 1.00 | 1.00 | 1.00 | 2 |
| 13 | 1.00 | 1.00 | 1.00 | 1 |
| 14 | 1.00 | 1.00 | 1.00 | 3 |
| 15 | 0.67 | 1.00 | 0.80 | 2 |
| 16 | 0.00 | 0.00 | 0.00 | 0 |
| 17 | 0.75 | 1.00 | 0.86 | 3 |
| 18 | 1.00 | 1.00 | 1.00 | 2 |
| 19 | 0.50 | 1.00 | 0.67 | 1 |
| 20 | 1.00 | 1.00 | 1.00 | 2 |
| 21 | 0.50 | 1.00 | 0.67 | 1 |
| 22 | 1.00 | 1.00 | 1.00 | 4 |
| 23 | 1.00 | 1.00 | 1.00 | 4 |
| 24 | 1.00 | 1.00 | 1.00 | 2 |
| 25 | 1.00 | 0.50 | 0.67 | 2 |
| 26 | 1.00 | 1.00 | 1.00 | 4 |
| 27 | 1.00 | 0.67 | 0.80 | 3 |
| 28 | 1.00 | 1.00 | 1.00 | 2 |
| 29 | 1.00 | 1.00 | 1.00 | 1 |
| 30 | 1.00 | 1.00 | 1.00 | 1 |
| 31 | 0.50 | 1.00 | 0.67 | 1 |
| 32 | 1.00 | 1.00 | 1.00 | 3 |
| 33 | 1.00 | 1.00 | 1.00 | 2 |
| 34 | 1.00 | 1.00 | 1.00 | 1 |
| 35 | 1.00 | 1.00 | 1.00 | 1 |
| 36 | 1.00 | 1.00 | 1.00 | 2 |
| 37 | 1.00 | 1.00 | 1.00 | 3 |
| 38 | 1.00 | 0.80 | 0.89 | 5 |
| 39 | 1.00 | 0.75 | 0.86 | 4 |
| accuracy | | | 0.88 | 100 |
| macro avg | 0.88 | 0.90 | 0.87 | 100 |
| weighted avg | 0.93 | 0.88 | 0.88 | 100 |

Too confusing right? But the good news is our model is 88% accurate!

# THE PROJECT 1.08

## Interpretation

- **Accuracy (Overall Performance):** The model correctly identified faces 88% of the time.
- **Precision (Accuracy per Class):** On average, when the model claims to identify a person, it is correct 88% of the time.
- **Recall (Sensitivity):** On average, the model captured 90% of all faces present in the dataset.
- **F1-Score (Balance of Precision and Recall):** The model achieves an 87% balance between precision and recall.
- **Support (Number of Instances per Class):** Indicates the number of instances (faces) for each identity.
- **Macro Average (Class Averages):** The average performance across all classes is 88%.
- **Weighted Average (Considering Class Imbalance):** Adjusts the average considering the number of instances in each class, emphasising larger classes.

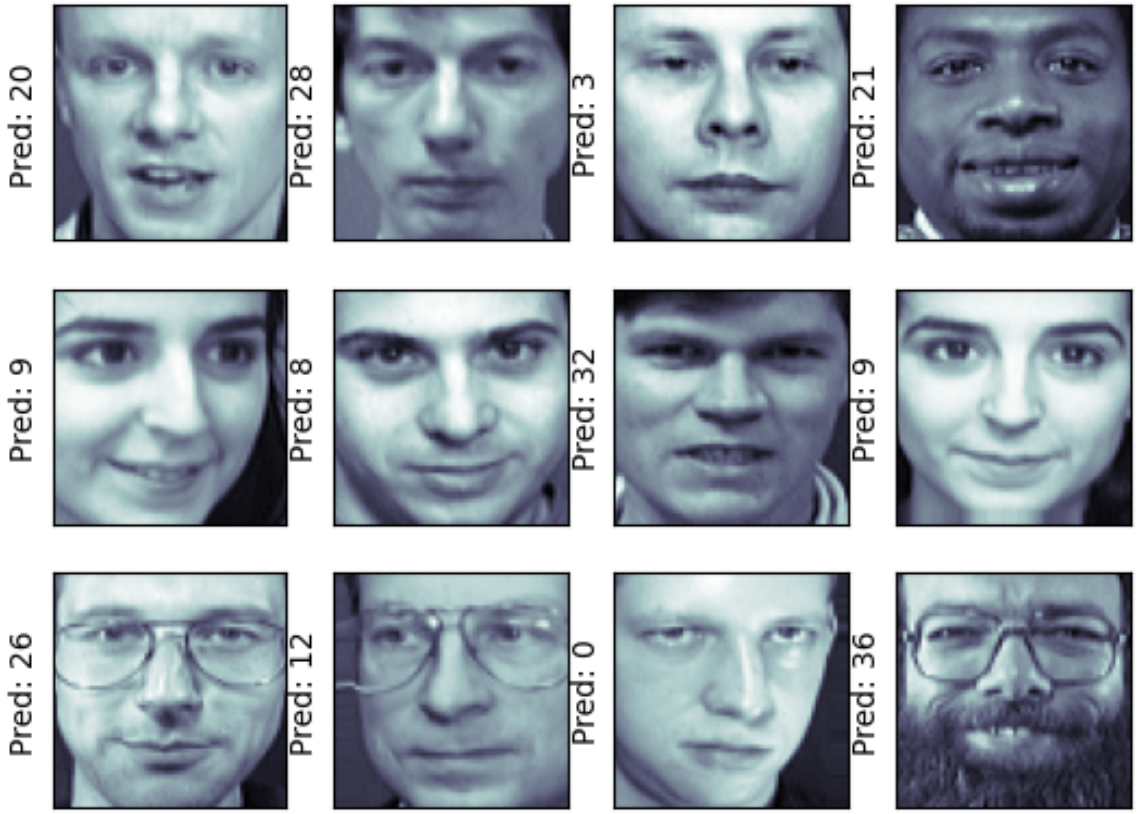
# THE PROJECT 1.09

## Or.... we can just visualise the results!

```
fig, ax = plt.subplots(3, 4) #visualizing and testing the model on the test set
for i, axi in enumerate(ax.flat):
    axi.imshow(X_test[i].reshape(64, 64), cmap='bone')
    axi.set(xticks=[], yticks=[])
    axi.set_ylabel(f'Pred: {y_pred[i]}', color='black')
plt.show()
```

- **fig, ax = plt.subplots(3, 4):** This line creates a 3x4 grid of subplots (12 plots in total) to visualize a subset of the test set.
- **for i, axi in enumerate(ax.flat):** It iterates through each subplot in the grid.
- **axi.imshow(X_test[i].reshape(64, 64), cmap='bone'):** It displays the image from the test set, reshaping it to the original 64x64 dimensions, using a grayscale color map ('bone' provides a black and white style).
- **axi.set(xticks=[ ], yticks=[ ]):** Removes the axis ticks for better visualization.
- **axi.set_ylabel(f'Pred: {y_pred[i]}', color='black'):** Adds a label indicating the predicted class for the displayed face. The label is set to black for visibility.
- **plt.show( ):** Finally, it shows the complete grid of subplots with the test images and their corresponding predicted labels

# THE PROJECT 1.09: THE OUTPUT!




If you take a look back when we visualise the dataset in 1.03,

The model has actually recognised every one successfully this time !

# CONCLUSION

- What is classification
- Importance of classification
- Types of classification
- Face classification
- The dataset
- Hands-on exercise / project

# Thanks for the attention!

VITTORIO STILE