Importing required libraries

In [1]:
```
import numpy as np
```

In [2]:
```
import matplotlib.pyplot as plt
```

In [29]:
```
from sklearn.datasets import load_digits
```

In [3]:
```
pip install scikit-learn matplotlib
```

```
Requirement already satisfied: scikit-learn in d:\jupyter\lib\site-packages
(1.3.0)
Requirement already satisfied: matplotlib in d:\jupyter\lib\site-packages (3.
7.2)
Requirement already satisfied: numpy>=1.17.3 in d:\jupyter\lib\site-packages
(from scikit-learn) (1.24.3)
Requirement already satisfied: scipy>=1.5.0 in d:\jupyter\lib\site-packages
(from scikit-learn) (1.11.1)
Requirement already satisfied: joblib>=1.1.1 in d:\jupyter\lib\site-packages
(from scikit-learn) (1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in d:\jupyter\lib\site-pa
ckages (from scikit-learn) (2.2.0)
Requirement already satisfied: contourpy>=1.0.1 in d:\jupyter\lib\site-packag
es (from matplotlib) (1.0.5)
Requirement already satisfied: cycler>=0.10 in d:\jupyter\lib\site-packages
(from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in d:\jupyter\lib\site-packa
ges (from matplotlib) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in d:\jupyter\lib\site-packa
ges (from matplotlib) (1.4.4)
Requirement already satisfied: packaging>=20.0 in d:\jupyter\lib\site-package
s (from matplotlib) (23.1)
Requirement already satisfied: pillow>=6.2.0 in d:\jupyter\lib\site-packages
(from matplotlib) (10.0.1)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in d:\jupyter\lib\site-p
ackages (from matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in d:\jupyter\lib\site-pa
ckages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in d:\jupyter\lib\site-packages (from
python-dateutil>=2.7->matplotlib) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

Find and load the Olivetti faces dataset

In [4]:
```
from sklearn.datasets import fetch_olivetti_faces
```

In [5]:
```
olivetti_faces = fetch_olivetti_faces(shuffle=True,random_state=42)
```

Description for this dataset

In [7]:
```
print("Description of Olivetti faces  dataset:")
print(olivetti_faces.DESCR)
```

```
Description of Olivetti faces  dataset:
.. _olivetti_faces_dataset:

The Olivetti faces dataset
--------------------------

`This dataset contains a set of face images`_ taken between April 1992 and
April 1994 at AT&T Laboratories Cambridge. The
:func:`sklearn.datasets.fetch_olivetti_faces` function is the data
fetching / caching function that downloads the data
archive from AT&T.

.. _This dataset contains a set of face images: https://cam-orl.co.uk/facedat
abase.html (https://cam-orl.co.uk/facedatabase.html)

As described on the original website:

    There are ten different images of each of 40 distinct subjects. For some
    subjects, the images were taken at different times, varying the lighting,
    facial expressions (open / closed eyes, smiling / not smiling) and facial
    details (glasses / no glasses). All the images were taken against a dark
    homogeneous background with the subjects in an upright, frontal position
    (with tolerance for some side movement).

**Data Set Characteristics:**

    =================   =====================
    Classes                                40
    Samples total                         400
    Dimensionality                       4096
    Features            real, between 0 and 1
    =================   =====================

The image is quantized to 256 grey levels and stored as unsigned 8-bit
integers; the loader will convert these to floating point values on the
interval [0, 1], which are easier to work with for many algorithms.

The "target" for this database is an integer from 0 to 39 indicating the
identity of the person pictured; however, with only 10 examples per class, th
is
relatively small dataset is more interesting from an unsupervised or
semi-supervised perspective.

The original dataset consisted of 92 x 112, while the version available here
consists of 64x64 images.

When using these images, please give credit to AT&T Laboratories Cambridge.
```

Targets

In [8]:
```
targets = olivetti_faces.target
print("\nTargets:")
print(targets)
```

```
Targets:
[20 28  3 21  9  8 32  9 26 12  0 36  5  7 13  4 27 37 23 38  7  1 39 27
  0 39 11 22 26 10 39 19 26  5 23 11 11 34 15 14 38  5  7  2  8 38 14 18
  2 17  4 32 33  7 37  3 22 17  3 15 12 29 25  7 10  3 35 26 39  7 32 14
  0  4 38 24 22 36 17 28  0  1 20 25 27  6 24 30 10  9 23 33 11 22 18 31
 37 38 23  7 24 11  1  6 15  0  1 13 35 34 13 38 29 38 29  6  7 28 30 28
 15 10  1 34  2 17 35 33 16 24 31 14 25 17 11 19 22 26 21 30  3 13 29 15
 19 28  5 11 16 36  0 33 27 15  1 19 10  8 31 39 37 20 28 16 35  8 37 16
 14 22  9  6 12  9 14 32  9 23  6  2  3 14 12 18  6 19 32 21 31 19 12 14
 37  8 33 34 33 35 33 30 18 20 28 21 28 12  3  1 32 18 22 11 17 32 29 11
 36 27 38 28 36 16 25 13 15 19 19 39  0 20 11 23 23  2 12 35 22 36 37 35
 37 12  7 32  2  8 38 10 24 29 13 24 18 29  4 36  6  8 24 18 15  1  3  2
 17 14 31 27 22  9  5 24 29 30 17  4 31 20 25 33  0 25 35 10 22 34 21 17
  9 21  6  4  3 26 20 35  2 31 23 26 28 16 37 13  6 13 12  0  6 30  1 15
  4 36 32 21 27 34 23 20 21 29 36 25 39 36 30 26 20 16  4 21 19 30 25 10
  5  0  4  8 20  3 26  9 33  5 34 26 24  1 31  8 27 16 32 39 13 30 38 31
 24  5  5 17 18 39 18 16  5  4 34 23 25  2 31 16 27 19 29 34 25 30 14 13
 15 35  9 37  8 33 21 12 39  2 18  7 10 27 34 10]
```

Features

In [9]:
```
features = olivetti_faces.data
print("\nFeatures:")
print(features)
```

```
Features:
[[0.1983471  0.23553719 0.35123968 ... 0.06198347 0.12809917 0.09090909]
 [0.18595041 0.12809917 0.11570248 ... 0.19008264 0.2107438  0.2107438 ]
 [0.5082645  0.60330576 0.6198347  ... 0.33471075 0.3429752  0.3429752 ]
 ...
 [0.61157024 0.6446281  0.6570248  ... 0.17768595 0.2107438  0.2231405 ]
 [0.28512397 0.29338843 0.29752067 ... 0.53305787 0.53305787 0.5371901 ]
 [0.3264463  0.1446281  0.2603306  ... 0.14049587 0.30165288 0.1570248 ]]
```

Data an labeled data

In [10]:
```
print("\nData Shape:")
print(features)
```

```
Data Shape:
[[0.1983471  0.23553719 0.35123968 ... 0.06198347 0.12809917 0.09090909]
 [0.18595041 0.12809917 0.11570248 ... 0.19008264 0.2107438  0.2107438 ]
 [0.5082645  0.60330576 0.6198347  ... 0.33471075 0.3429752  0.3429752 ]
 ...
 [0.61157024 0.6446281  0.6570248  ... 0.17768595 0.2107438  0.2231405 ]
 [0.28512397 0.29338843 0.29752067 ... 0.53305787 0.53305787 0.5371901 ]
 [0.3264463  0.1446281  0.2603306  ... 0.14049587 0.30165288 0.1570248 ]]
```

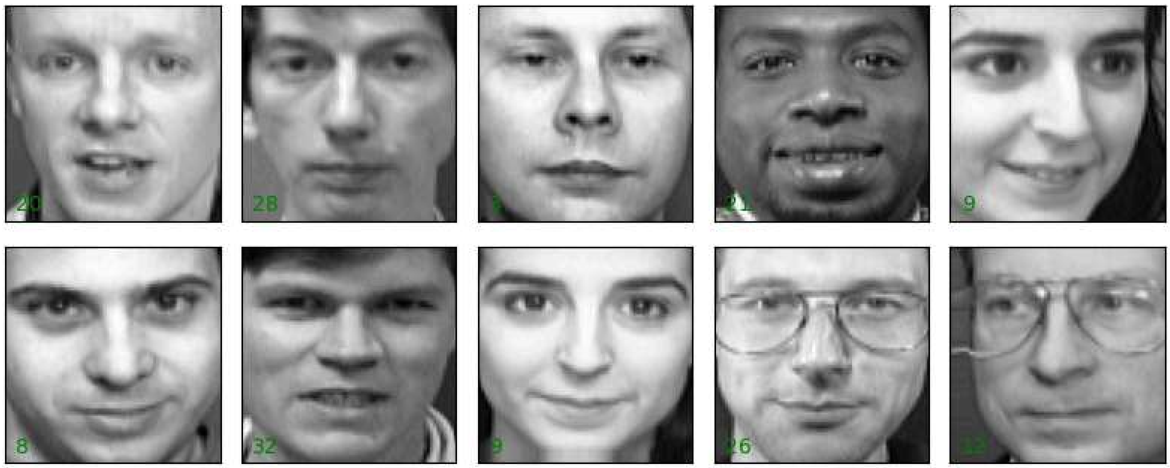
To print Shape and datatype of various items

In [12]:
```
print("\nTarget Shape:")
print(targets.shape)
print("\nFeatures Datatype:")
print(features.dtype)
```

```
Target Shape:
(400,)

Features Datatype:
float32
```

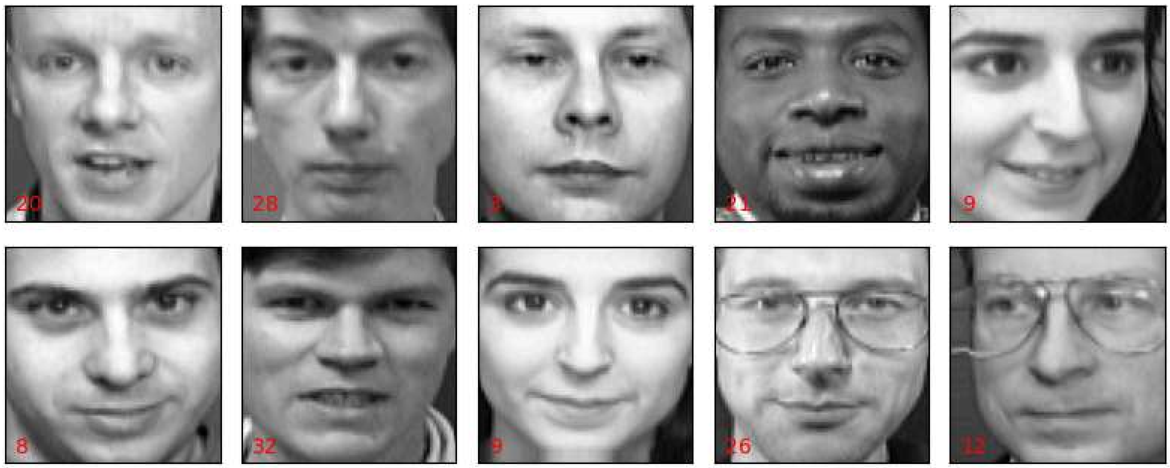
```
Display varios items
```

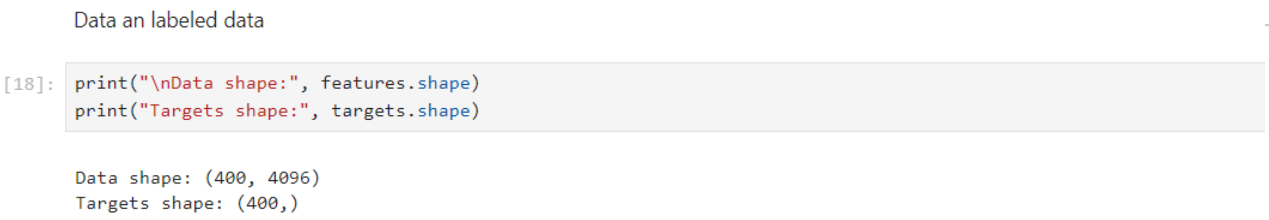
In [32]:
```
fig, axes = plt.subplots(2, 5, figsize=(10, 4),
                         subplot_kw={'xticks':[], 'yticks':[]},
                         gridspec_kw=dict(hspace=0.1, wspace=0.1))

for i, ax in enumerate(axes.flat):
    ax.imshow(features[i].reshape(64, 64), cmap='gray')
    ax.text(0.05, 0.05, str(targets[i]), transform=ax.transAxes, color='green'

plt.show()
```



plotting the digits and labeling the image with the target value

```
In [27]: fig, axes = plt.subplots(2,5,figsize=(10,4),
                                  subplot_kw={'xticks':[],'yticks':[]},
                                  gridspec_kw=dict(hspace=0.1,wspace=0.1))
         for i, ax in enumerate(axes.flat):
             ax.imshow(features[i].reshape(64,64),cmap='gray')
             ax.text(0.05,0.05,str(targets[i]),transform=ax.transAxes,
                     color='green' if targets[i] == 1 else 'red')
         plt.show()
```





Data an labeled data

```
[18]: print("\nData shape:", features.shape)
      print("Targets shape:", targets.shape)

Data shape: (400, 4096)
Targets shape: (400,)
```