In [1]:

```
#Ayush Sharma 209303312
# Python code to implement Principal component analysis for dimensionality reductio
from sklearn.datasets import load_digits
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.decomposition import PCA
```

In [2]:

```
dataset = load_digits()
dataset.keys()
```

Out[2]:

```
dict_keys(['data', 'target', 'frame', 'feature_names', 'target_names', 'images',
'DESCR'])
```

In [3]:

```
print(dataset.data.shape)
dataset.data[0]
```

```
(1797, 64)
```

Out[3]:

```
array([ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.,  0.,  0., 13., 15., 10.,
       15.,  5.,  0.,  0.,  3., 15.,  2.,  0., 11.,  8.,  0.,  0.,  4.,
       12.,  0.,  0.,  8.,  8.,  0.,  0.,  5.,  8.,  0.,  0.,  9.,  8.,
        0.,  0.,  4., 11.,  0.,  1., 12.,  7.,  0.,  0.,  2., 14.,  5.,
       10., 12.,  0.,  0.,  0.,  0.,  6., 13., 10.,  0.,  0.,  0.])
```

In [4]:

```
dataset.data[0].reshape(8,8)
```

Out[4]:

```
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

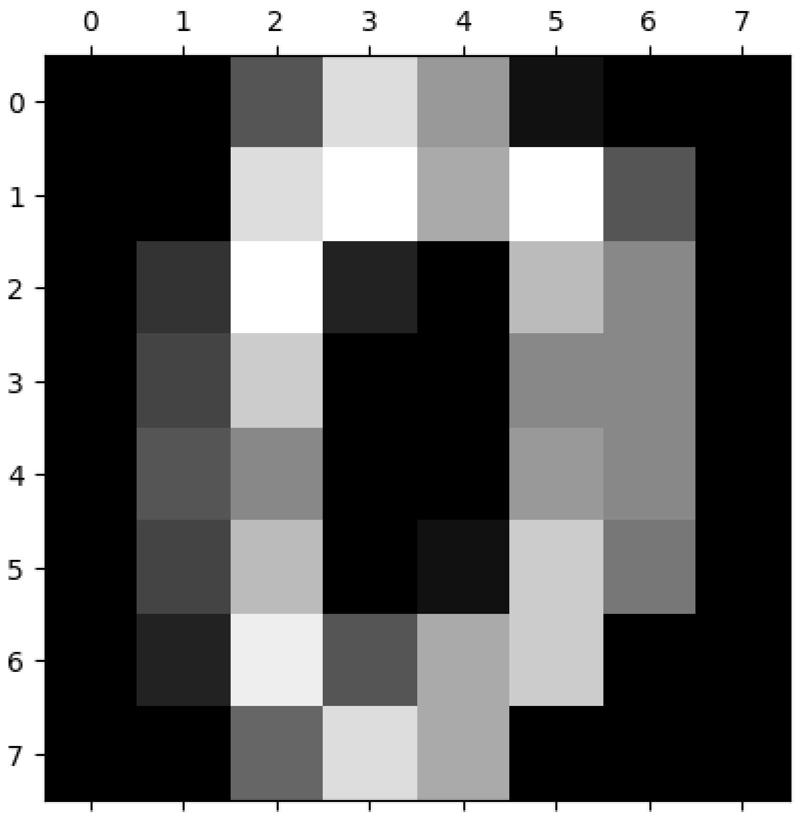
In [5]:

```
plt.gray()
plt.matshow(dataset.data[0].reshape(8,8))
```

Out[5]:

```
<matplotlib.image.AxesImage at 0x1e8e34ef1f0>
```

```
<Figure size 640x480 with 0 Axes>
```



```
In [6]: dataset.target[:5]
```

```
Out[6]: array([0, 1, 2, 3, 4])
```

```
In [7]: df = pd.DataFrame(dataset.data, columns=dataset.feature_names)
        df.head()
        X = df
        y = dataset.target
```

```
In [8]: scaler = StandardScaler()
        X_scaled = scaler.fit_transform(X)
        X_scaled
```

```
Out[8]: array([[ 0.        , -0.33501649, -0.04308102, ..., -1.14664746,
                -0.5056698 , -0.19600752],
               [ 0.        , -0.33501649, -1.09493684, ...,  0.54856067,
                -0.5056698 , -0.19600752],
               [ 0.        , -0.33501649, -1.09493684, ...,  1.56568555,
                 1.6951369 , -0.19600752],
               ...,
               [ 0.        , -0.33501649, -0.88456568, ..., -0.12952258,
                -0.5056698 , -0.19600752],
               [ 0.        , -0.33501649, -0.67419451, ...,  0.8876023 ,
                -0.5056698 , -0.19600752],
               [ 0.        , -0.33501649,  1.00877481, ...,  0.8876023 ,
                -0.26113572, -0.19600752]])
```

```
In [9]: X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, ran
        model = LogisticRegression()
        model.fit(X_train, y_train)
```

```
Out[9]:  ▾ LogisticRegression
         LogisticRegression()
```

```
In [10]:  model.score(X_test, y_test)
```

```
Out[10]:  0.9722222222222222
```

```
In [11]:  pca = PCA(0.95)
          X_pca = pca.fit_transform(X)
          X_pca.shape
```

```
Out[11]:  (1797, 29)
```

```
In [12]:  X_train_pca, X_test_pca, y_train, y_test = train_test_split(X_pca, y, test_size=0.2
          model = LogisticRegression(max_iter=1000)
          model.fit(X_train_pca, y_train)
          model.score(X_test_pca, y_test)
```

```
Out[12]:  0.9694444444444444
```

```
In [13]:  pca = PCA(n_components=2)
          X_pca = pca.fit_transform(X)
          X_pca.shape
```

```
Out[13]:  (1797, 2)
```

```
In [14]:  X_train_pca, X_test_pca, y_train, y_test = train_test_split(X_pca, y, test_size=0.2
          model = LogisticRegression(max_iter=1000)
          model.fit(X_train_pca, y_train)
          model.score(X_test_pca, y_test)
```

```
Out[14]:  0.6083333333333333
```