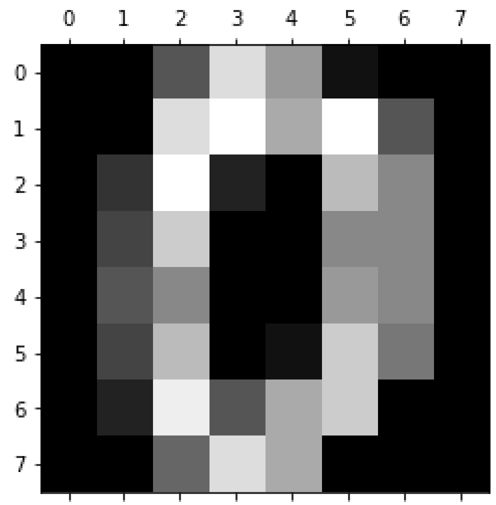
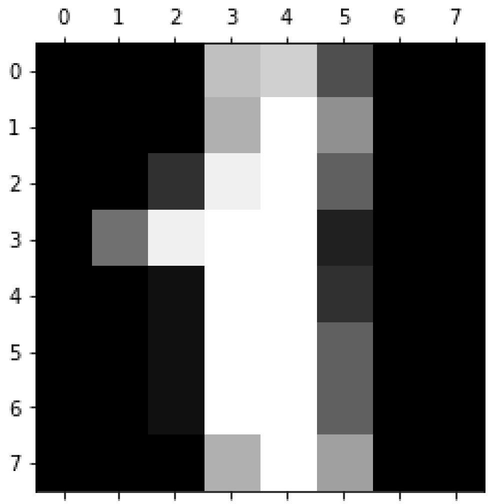
Logistic Regression: Multiclass Classification

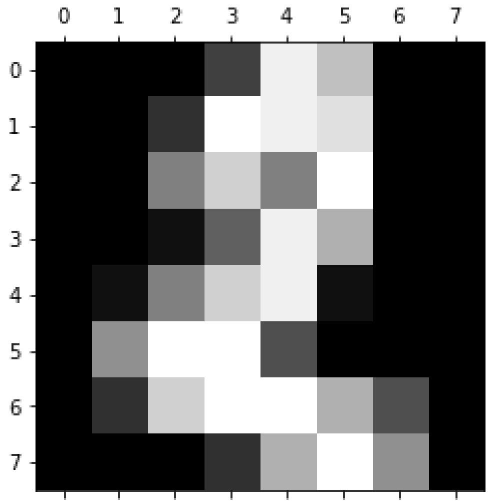
In [1]:
```
from sklearn.datasets import load_digits
%matplotlib inline
import matplotlib.pyplot as plt
digits = load_digits()
```

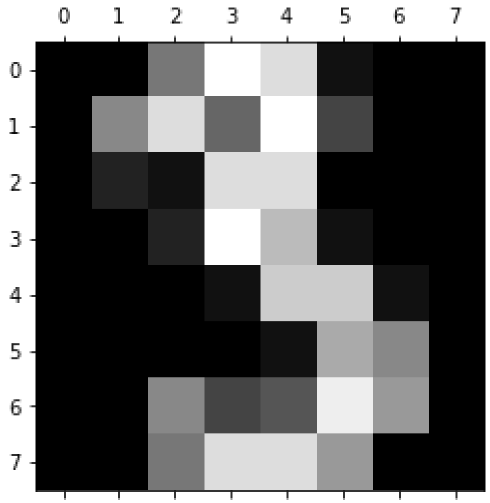
In [2]:
```
plt.gray()
for i in range(5):
    plt.matshow(digits.images[i])
```

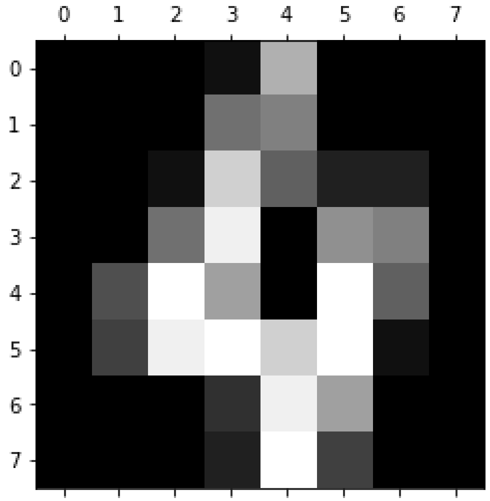
```
<Figure size 432x288 with 0 Axes>
```




0
1
2
3
4
5
6
7
0
1
2
3
4
5
6
7


0
1
2
3
4
5
6
7
0
1
2
3
4
5
6
7





In [3]:
```
dir(digits)
```

Out[3]:
```
['DESCR', 'data', 'images', 'target', 'target_names']
```

In [4]:
```
digits.data[0]
```

```
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.])
```

Create and train logistic regression model

```
In [5]: from sklearn.linear_model import LogisticRegression
        model = LogisticRegression()
```

```
In [6]: from sklearn.model_selection import train_test_split
```

```
In [15]: X_train, X_test, y_train, y_test = train_test_split(digits.data,digits.
         target, test_size=0.2)
```

```
In [22]: len(X_train)
```

```
Out[22]: 1437
```

```
In [23]: len(X_test)
```

```
Out[23]: 360
```

```
In [24]: model.fit(X_train, y_train)
```

```
C:\Users\prasa\anaconda3\lib\site-packages\sklearn\linear_model\_logist
ic.py:940: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

```
Out[24]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=
True,
                            intercept_scaling=1, l1_ratio=None, max_iter=100,
                            multi_class='auto', n_jobs=None, penalty='l2',
                            random_state=None, solver='lbfgs', tol=0.0001, verbo
se=0,
                            warm_start=False)
```

Measure accuracy of our model

```
In [25]: model.score(X_test, y_test)
```

```
Out[25]: 0.9666666666666667
```

```
In [26]: model.predict(digits.data[0:5])
```

```
Out[26]: array([0, 1, 2, 3, 4])
```

Confusion Matrix

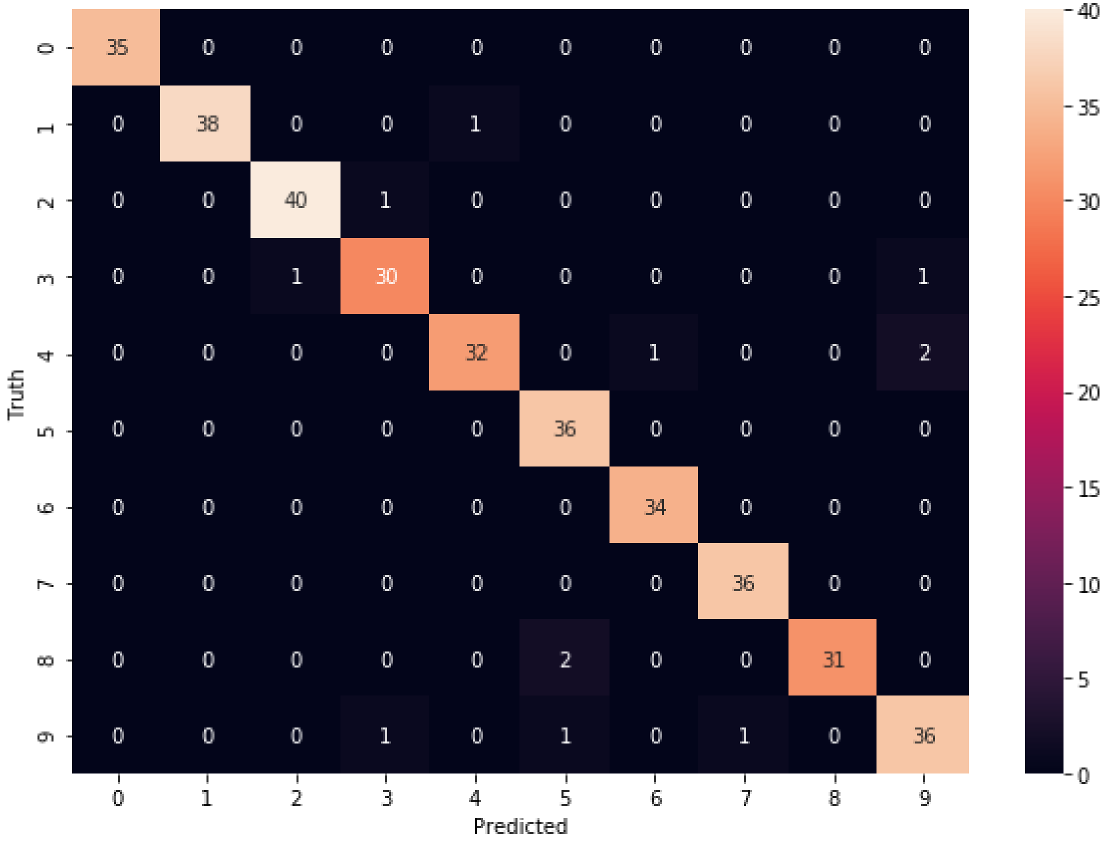
```
In [30]: y_predicted = model.predict(X_test)
```

```
In [31]: from sklearn.metrics import confusion_matrix
         cm = confusion_matrix(y_test,y_predicted)
         cm
```

```
Out[31]: array([[35,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0, 38,  0,  0,  1,  0,  0,  0,  0,  0],
                [ 0,  0, 40,  1,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  1, 30,  0,  0,  0,  0,  0,  1],
                [ 0,  0,  0,  0, 32,  0,  1,  0,  0,  2],
                [ 0,  0,  0,  0,  0, 36,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0, 34,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  0,  0, 36,  0,  0],
                [ 0,  0,  0,  0,  0,  2,  0,  0, 31,  0],
                [ 0,  0,  0,  1,  0,  1,  0,  1,  0, 36]], dtype=int64)
```

In [33]:
```python
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm,annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

Out[33]:
```
Text(69.0, 0.5, 'Truth')
```



Exercise

Use sklearn.datasets iris flower dataset to train your model using logistic regression. You need to figure out accuracy of your model and use that to predict different samples in your test dataset. In iris dataset there are 150 samples containing following features,

1. Sepal Length
2. Sepal Width
3. Petal Length
4. Petal Width

Using above 4 features you will clasify a flower in one of the three categories,

1. Setosa
2. Versicolour
3. Virginica