Logistic regression:

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary).

Like all regression analyses, logistic regression is a predictive analysis.

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable

and one or more nominal, ordinal, interval or ratio-level independent variables.

How does Logistic Regression work?

- Logistic regression works in the following steps:

1, Prepare the data:

- The data should be in a format where each row represents a single observation
- and each column represents a different variable. The target variable (the variable you want to predict)
- should be binary (yes/no, true/false, 0/1).

2,Train the model:

- We teach the model by showing it the training data.
- This involves finding the values of the model parameters that minimize the error in the training data.

3, Evaluate the model:

- The model is evaluated on the held-out test data to assess its performance on unseen data.
- Use the model to make predictions: After the model has been trained and assessed,
- it can be used to forecast outcomes on new data.

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

import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
digits = load_digits()
```

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

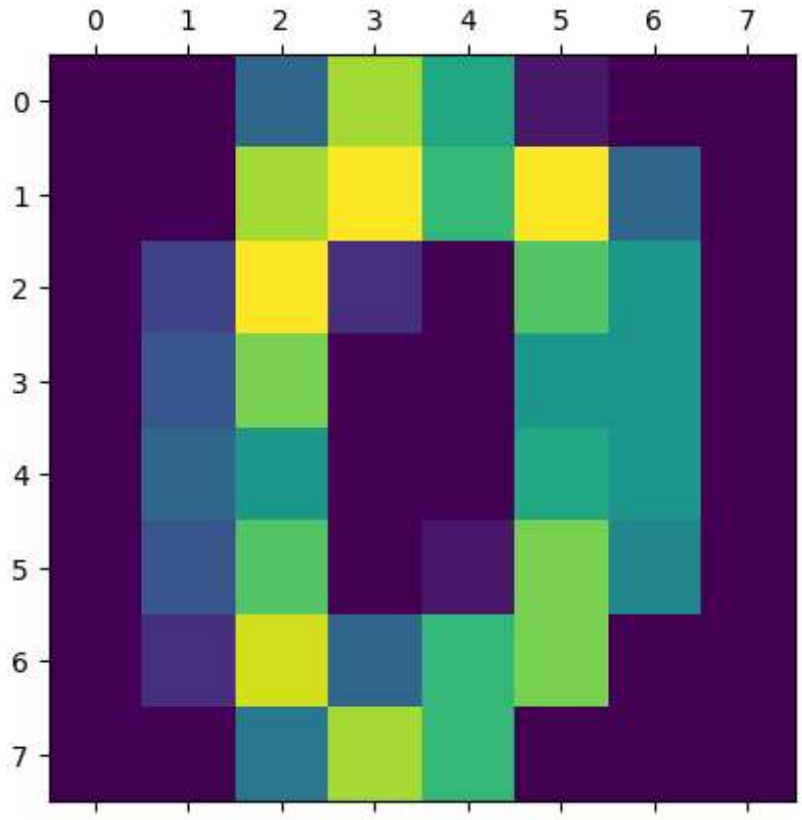
Out[3]:
```
['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_name
s']
```

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

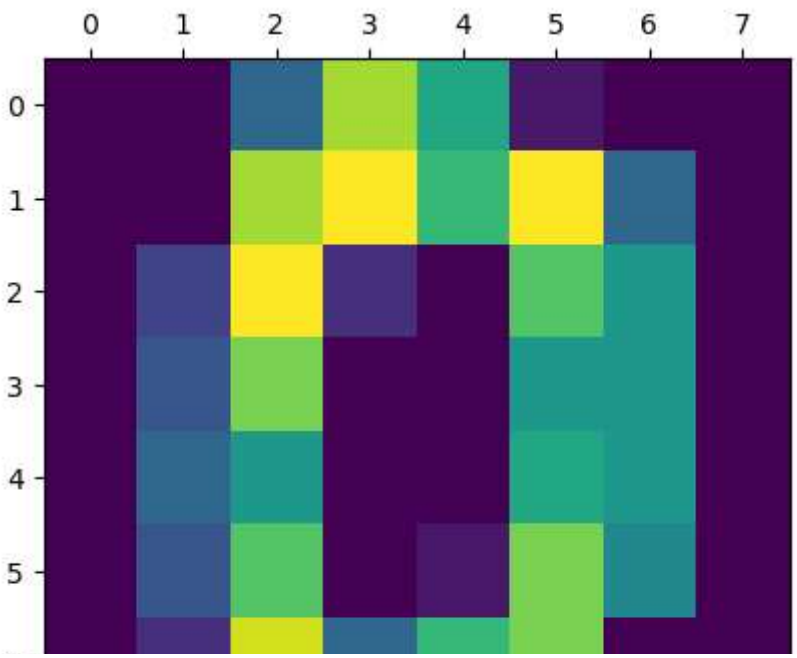
Out[4]:
```
array([ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.,  0.,  0., 13., 15., 10.,
       15.,  5.,  0.,  0.,  3., 15.,  2.,  0., 11.,  8.,  0.,  0.,  4.,
       12.,  0.,  0.,  8.,  8.,  0.,  0.,  5.,  8.,  0.,  0.,  9.,  8.,
        0.,  0.,  4., 11.,  0.,  1., 12.,  7.,  0.,  0.,  2., 14.,  5.,
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

In [5]:
```
plt.gray
plt.matshow(digits.images[0])
```

Out[5]:
```
<matplotlib.image.AxesImage at 0x1a9ad456c20>
```



In [6]:
```
plt.gray
for i in range(6):
    plt.matshow(digits.images[i])
```



In [7]:
```
digits.target[0:5]
```

Out[7]:
```
array([0, 1, 2, 3, 4])
```

In [8]:
```
# Split the data into a training set and a test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(digits.data ,digits.ta
```

In [9]:
```
len(X_train)
```

Out[9]:
```
1437
```

In [10]:
```
len(X_test)
```

Out[10]:
```
360
```

In [11]:
```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()

```

In [12]:
```
model.fit(X_train, y_train)
```

Out[12]:
```
LogisticRegression
LogisticRegression()
```

In [13]:
```
model.score(X_test, y_test)
```

Out[13]:
```
0.9694444444444444
```

In [14]:
```
model.predict(digits.data[0:5])
```

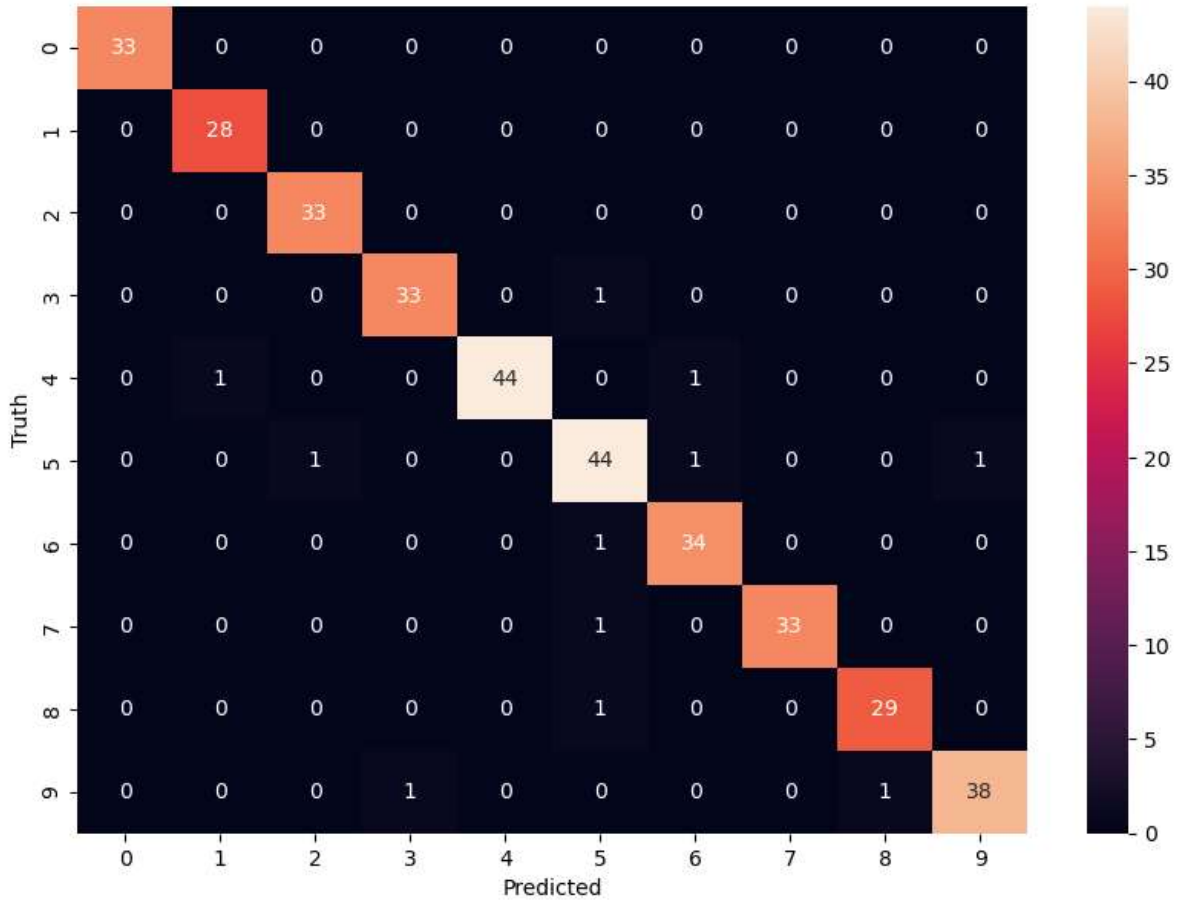
Out[14]:
```
array([0, 1, 2, 3, 4])
```

In [15]:
```
y_predicted =model.predict(X_test)
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_predicted)
cm
```

Out[15]:
```
array([[33,  0,  0,  0,  0,  0,  0,  0,  0,  0],
       [ 0, 28,  0,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0, 33,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0,  0, 33,  0,  1,  0,  0,  0,  0],
       [ 0,  1,  0,  0, 44,  0,  1,  0,  0,  0],
       [ 0,  0,  1,  0,  0, 44,  1,  0,  0,  1],
       [ 0,  0,  0,  0,  0,  1, 34,  0,  0,  0],
       [ 0,  0,  0,  0,  0,  1,  0, 33,  0,  0],
       [ 0,  0,  0,  0,  0,  1,  0,  0, 29,  0],
       [ 0,  0,  0,  1,  0,  0,  0,  0,  1, 38]], dtype=int64)
```

In [16]:
```
import seaborn as sns

plt.figure(figsize = (10, 7))
sns.heatmap(cm , annot = True)
plt.xlabel("Predicted")
plt.ylabel("Truth")
```

Out[16]: Text(95.72222222222221, 0.5, 'Truth')



In [ ]:
```
1
```