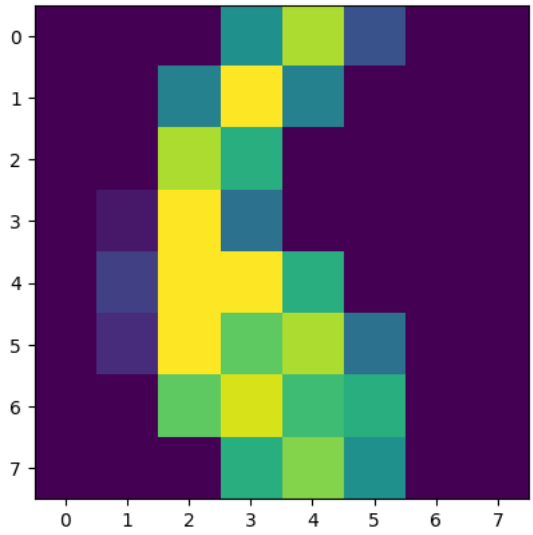
```
import numpy as np
import matplotlib.pyplot as plot
%matplotlib inline

import sklearn
from sklearn.datasets import  load_digits

digits = load_digits()

X = digits.data
y = digits.target



print("Shape of X is {}".format(X.shape))
print("Shape of y is {}".format(y.shape))
X[0]
```

```
Shape of X is (1797, 64)
Shape of y is (1797,)
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.])
```

```
def plot_digit(x,index):
    plot.imshow(x.reshape(8,8))
    print(index)
```

```
plot_digit(X[104],y[104])
```

```
6
```




```
plot_digit(X[122],y[122])
```

```
8
```



```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
print(X_train.shape)
```

```
(1437, 64)
```



```
from sklearn.linear_model import LogisticRegression
```

Lets build a logistic regression model and test its accuracy for X_train (already seen) data

```
5
```

```
lr = LogisticRegression()
lr.fit(X_train,y_train)
y_predict1 = lr.predict(X_train)
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_predict1,y_train)
print(accuracy)
```

```
1.0
/usr/local/lib/python3.9/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=
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
  n_iter_i = _check_optimize_result(
```

Lets use the same model and test its accuracy for X_test (unseen) data

```
y_predict = lr.predict(X_test)
accuracy = accuracy_score(y_predict,y_test)
print(accuracy)
```

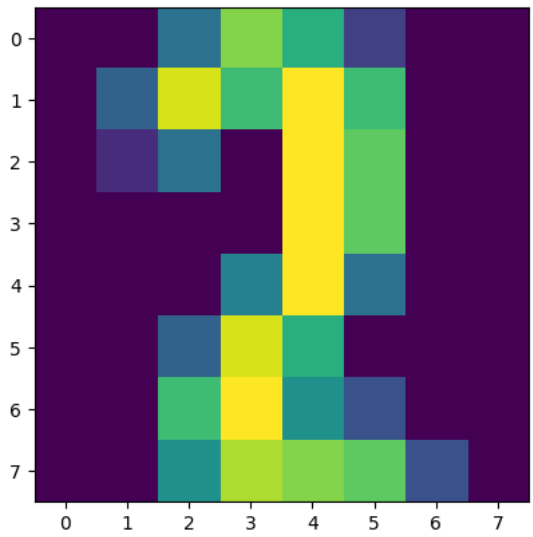
```
0.9694444444444444
```

```
lr.predict([X[100], X[152]])
```

```
array([4, 2])
```

```
plot_digit(X[152],y[152])
```

```
2
```



✓ 0s completed at 11:43 AM