```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn import metrics

# Load the digits dataset
digits = datasets.load_digits()

# Split the data into features (X) and labels (y)
X = digits.data
y = digits.target
"""
print(X)
Output :
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0. 10. ... 12.  1.  0.]]
"""
# print(y)   // 0 1 2 ... 8 9 8
```

{"type":"string"}

```python
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Create an SVM classifier (linear kernel)
clf = svm.SVC(kernel='linear')

# Fit the classifier on the training data
clf.fit(X_train, y_train)
```

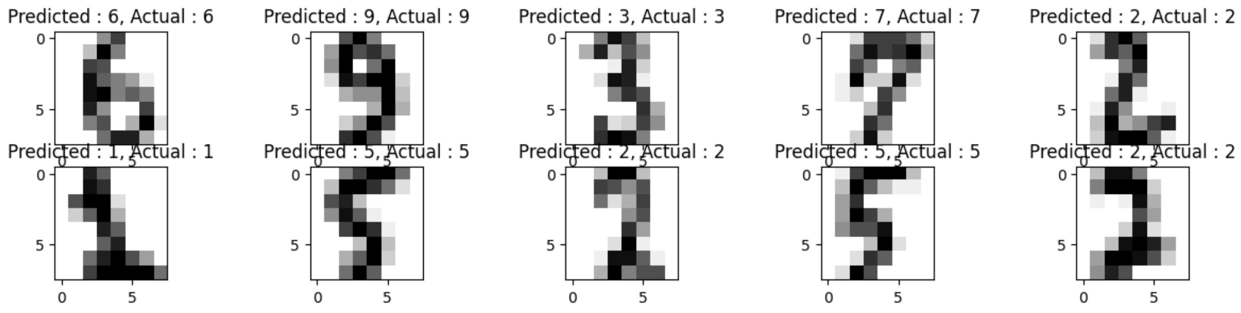
```
SVC(kernel='linear')
```

```python
# Predict on the test data
y_pred = clf.predict(X_test)

# Calculate accuracy
accuracy = metrics.accuracy_score(y_test, y_pred)
print("Accuracy : ", accuracy)
```

```
Accuracy :  0.9777777777777777
```

```python
# Confusion matrix
confusion_matrix = metrics.confusion_matrix(y_test, y_pred)
print("Confusion Matrix : ")
print(confusion_matrix)
```

```
Confusion Matrix :
[[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 32  0  1  0  0  0  1]
 [ 0  1  0  0 45  0  0  0  0  0]
 [ 0  0  0  0  0 47  0  0  0  0]
 [ 0  0  0  0  0  0 35  0  0  0]
 [ 0  0  0  0  0  0  0 33  0  1]
 [ 0  0  0  0  0  1  0  0 29  0]
 [ 0  0  0  1  1  0  0  1  0 37]]

# Classification report
classification_report = metrics.classification_report(y_test, y_pred)
print("Classification Report : ")
print(classification_report)

Classification Report :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        33
           1       0.97      1.00      0.98        28
           2       1.00      1.00      1.00        33
           3       0.97      0.94      0.96        34
           4       0.98      0.98      0.98        46
           5       0.96      1.00      0.98        47
           6       1.00      1.00      1.00        35
           7       0.97      0.97      0.97        34
           8       1.00      0.97      0.98        30
           9       0.95      0.93      0.94        40

    accuracy                           0.98       360
   macro avg       0.98      0.98      0.98       360
weighted avg       0.98      0.98      0.98       360


# Visualize some of the test images and their predicted labels
plt.figure(figsize=(15, 8))
for i in range(10):
    plt.subplot(5, 5, i + 1)
    plt.imshow(X_test[i].reshape(8, 8), cmap=plt.cm.gray_r)
    plt.title(f"Predicted : {y_pred[i]}, Actual : {y_test[i]}")
    plt.axis('on')
```


Predicted : 6, Actual : 6
Predicted : 9, Actual : 9
Predicted : 3, Actual : 3
Predicted : 7, Actual : 7
Predicted : 2, Actual : 2
Predicted : 1, Actual : 1
Predicted : 5, Actual : 5
Predicted : 2, Actual : 2
Predicted : 5, Actual : 5
Predicted : 2, Actual : 2