Support Vector Machine

```
from sklearn.datasets import load_digits
digits= load_digits()
dir(digits)
```

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

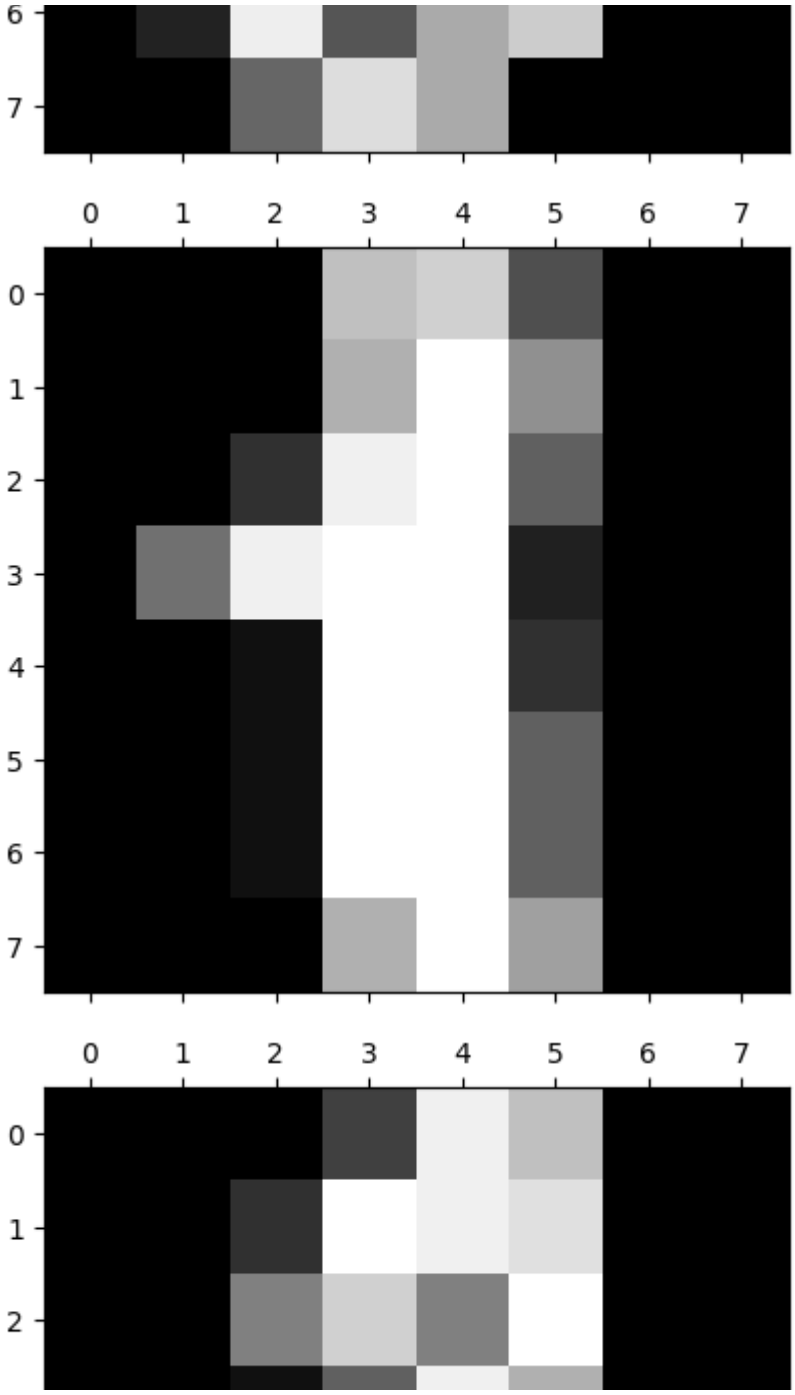
```
print(digits.target_names)
```

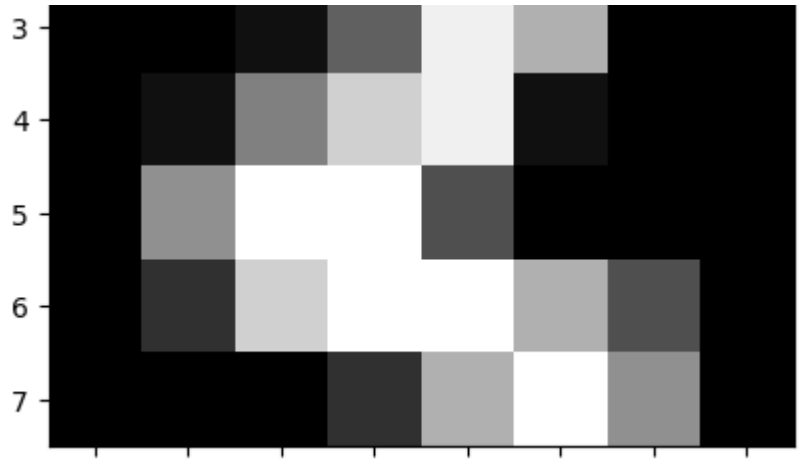
```
[0 1 2 3 4 5 6 7 8 9]
```

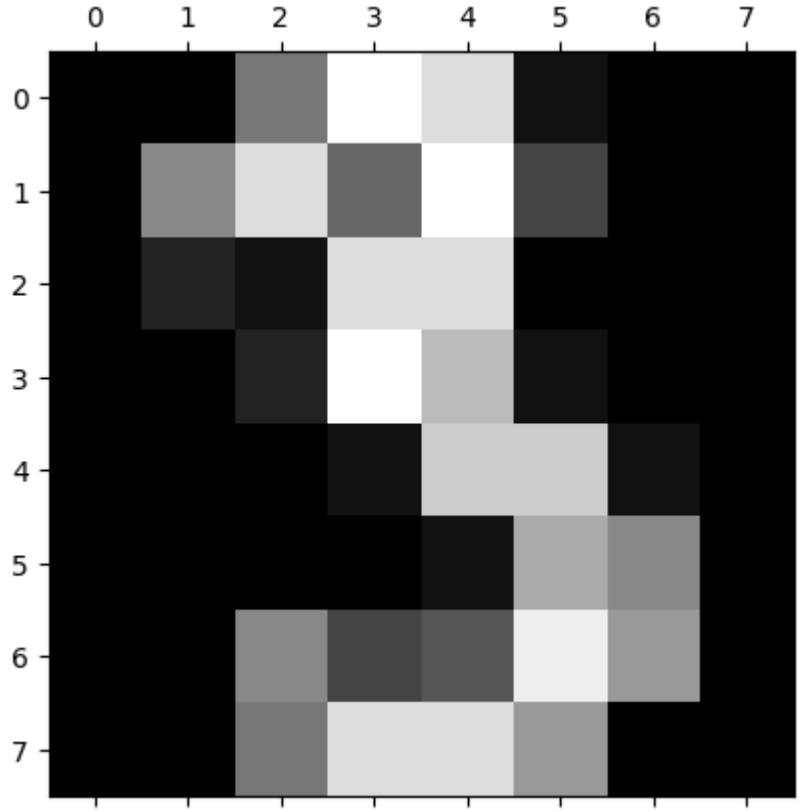
```
print(digits.data[0])
```

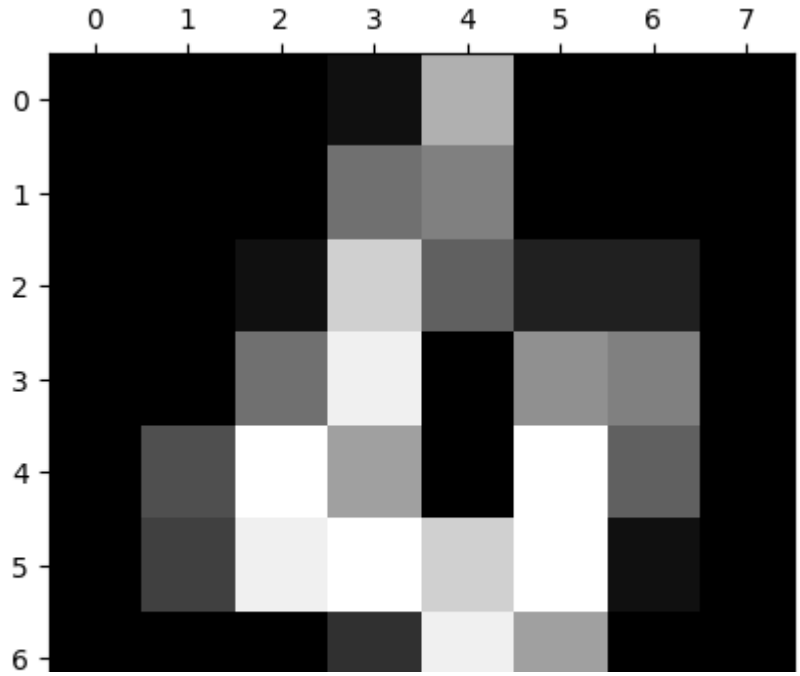
```
[ 0.  0.  5. 13.  9.  1.  0.  0.  0.  0. 13. 15. 10. 15.  5.  0.  0.  3.
 15.  2.  0. 11.  8.  0.  0.  4. 12.  0.  0.  8.  8.  0.  0.  5.  8.  0.
  0.  9.  8.  0.  0.  4. 11.  0.  1. 12.  7.  0.  0.  2. 14.  5. 10. 12.
  0.  0.  0.  0.  6. 13. 10.  0.  0.  0.]
```

```
import matplotlib.pyplot as plt
plt.gray()
for i in range(5):
    plt.matshow(digits.images[i])
```


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
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
0
1
2
3
4
5
6

```
import pandas as pd
df= pd.DataFrame(digits.data,columns=digits.feature_names)
df.head()
```

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0

5 rows × 64 columns

```
target= pd.DataFrame(digits.target,columns=['target'])
dataset= pd.concat([df,target],axis='columns')
dataset.head()
```

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0

5 rows × 65 columns

```
dataset.isna().sum()
```

	0
pixel_0_0	0
pixel_0_1	0
pixel_0_2	0
pixel_0_3	0
pixel_0_4	0
...	...
pixel_7_4	0
pixel_7_5	0
pixel_7_6	0
pixel_7_7	0
target	0

65 rows × 1 columns

dtype: int64

```
dataset.shape
```

```
(1797, 65)
```

```
x=dataset.drop('target',axis='columns')
y=dataset.target
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=55,test_size=0.2)
```

```
from sklearn.svm import SVC
model=SVC(kernel='poly')
model.fit(x_train,y_train)
```

```
SVC(kernel='poly')
```

```
model.score(x_test,y_test)
```

```
0.9944444444444445
```

RANDOM FOREST

```
from sklearn.ensemble import RandomForestClassifier
model1=RandomForestClassifier(n_estimators=40)
model1.fit(x_train,y_train)
```

```
RandomForestClassifier
RandomForestClassifier(n_estimators=40)
```

```
model1.score(x_test,y_test)
```

```
0.9833333333333333
```

```
y_predict=model1.predict(x_test)
y_pred=model.predict(x_test)
```

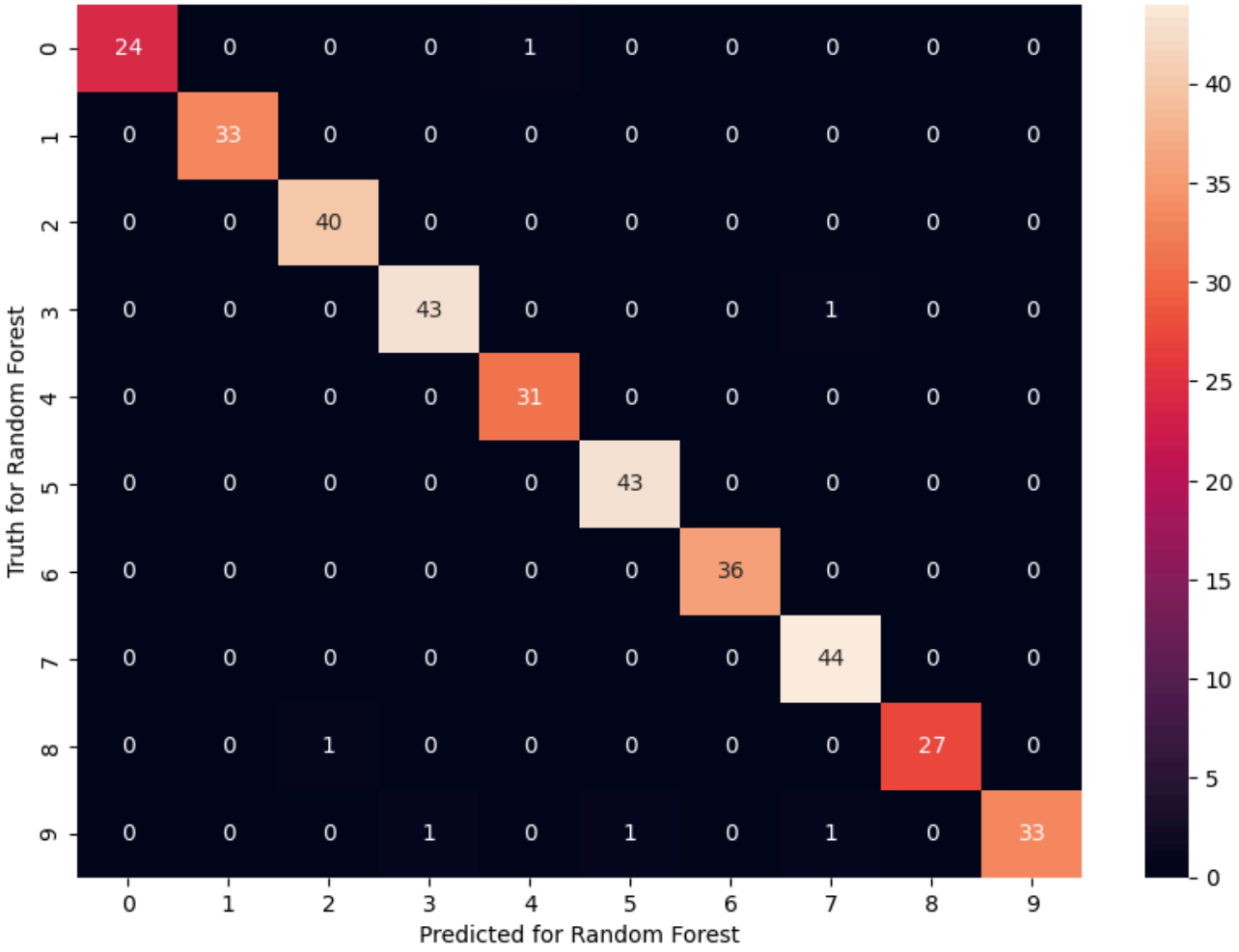
```
from sklearn.metrics import confusion_matrix
cm1=confusion_matrix(y_test,y_predict)
cm=confusion_matrix(y_test,y_pred)
```

```
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10,7))
sns.heatmap(cm,annot=True)
plt.xlabel('Predicted for SVC')
plt.ylabel('Truth for SVC')
```

```
Text(95.72222222222221, 0.5, 'Truth for SVC')
```


```
plt.figure(figsize=(10,7))
sns.heatmap(cm1,annot=True)
plt.xlabel('Predicted for Random Forest')
plt.ylabel('Truth for Random Forest')
```

```
Text(95.72222222222221, 0.5, 'Truth for Random Forest')
```



PCA

```
from sklearn.decomposition import PCA
```