CL1-05

July 24, 2025

[ ]:
```
'''NAME:Aher Swami Sandip
ROLL NO.01
COURSE: AI&DS
CLASS: BE
SUB:Computer Laboratory-I (Machine Learning)'''
```

[ ]:
```
'''
PRACTICAL NO-05:
    Ensemble Learning
Use different voting mechanism and Apply AdaBoost (Adaptive Boosting), Gradient
Tree Boosting (GBM), XGBoost classification on Iris dataset and compare the
performance of three models using different evaluation measures.
'''
```

[1]:
```
import pandas as pd
from sklearn.datasets import load_digits
digits = load_digits()
```

[2]:
```
dir(digits)
```

[2]:
```
['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_names']
```

[5]:
```
%matplotlib inline
import matplotlib.pyplot as plt
```

[7]:
```
plt.gray()
for i in range(4) :
    plt.matshow(digits.images[i])
```

[9]:
```
df = pd.DataFrame(digits.data)
df.head()
```

[9]:

	0	1	2	3	4	5	6	7	8	9	…	54	55	56	\
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	…	5.0	0.0	0.0	
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	…	9.0	0.0	0.0	

```
4  0.0  0.0  0.0   1.0  11.0   0.0  0.0  0.0  0.0  0.0  …  0.0  0.0  0.0

    57   58    59    60    61   62   63
0  0.0  6.0  13.0  10.0   0.0  0.0  0.0
1  0.0  0.0  11.0  16.0  10.0  0.0  0.0
2  0.0  0.0   3.0  11.0  16.0  9.0  0.0
3  0.0  7.0  13.0  13.0   9.0  0.0  0.0
4  0.0  0.0   2.0  16.0   4.0  0.0  0.0

[5 rows x 64 columns]
```

[11]:
```
df['target'] = digits.target
df[0: 12]
```

[11]:
```
      0    1     2     3     4     5     6    7    8    9  …   55   56   57  \
0   0.0  0.0   5.0  13.0   9.0   1.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
1   0.0  0.0   0.0  12.0  13.0   5.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
2   0.0  0.0   0.0   4.0  15.0  12.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
3   0.0  0.0   7.0  15.0  13.0   1.0   0.0  0.0  0.0  8.0  …  0.0  0.0  0.0
4   0.0  0.0   0.0   1.0  11.0   0.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
5   0.0  0.0  12.0  10.0   0.0   0.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
6   0.0  0.0   0.0  12.0  13.0   0.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
7   0.0  0.0   7.0   8.0  13.0  16.0  15.0  1.0  0.0  0.0  …  0.0  0.0  0.0
8   0.0  0.0   9.0  14.0   8.0   1.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
9   0.0  0.0  11.0  12.0   0.0   0.0   0.0  0.0  0.0  2.0  …  0.0  0.0  0.0
10  0.0  0.0   1.0   9.0  15.0  11.0   0.0  0.0  0.0  0.0  …  0.0  0.0  0.0
11  0.0  0.0   0.0   0.0  14.0  13.0   1.0  0.0  0.0  0.0  …  0.0  0.0  0.0

      58    59    60    61   62   63  target
0    6.0  13.0  10.0   0.0  0.0  0.0       0
1    0.0  11.0  16.0  10.0  0.0  0.0       1
2    0.0   3.0  11.0  16.0  9.0  0.0       2
3    7.0  13.0  13.0   9.0  0.0  0.0       3
4    0.0   2.0  16.0   4.0  0.0  0.0       4
5    9.0  16.0  16.0  10.0  0.0  0.0       5
6    1.0   9.0  15.0  11.0  3.0  0.0       6
7   13.0   5.0   0.0   0.0  0.0  0.0       7
8   11.0  16.0  15.0  11.0  1.0  0.0       8
9    9.0  12.0  13.0   3.0  0.0  0.0       9
10   1.0  10.0  13.0   3.0  0.0  0.0       0
11   0.0   1.0  13.0  16.0  1.0  0.0       1

[12 rows x 65 columns]
```

[13]:
```
# Train the model and Prediction
X = df.drop('target', axis = 'columns')
y = df.target
```

[15]:
```
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)
```

[17]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators = 20)
model.fit(X_train, y_train)
```

[17]:
```
RandomForestClassifier(n_estimators=20)
```

[19]:
```
RandomForestClassifier(n_estimators = 20)
model.score(X_test, y_test)
```

[19]:
```
0.9722222222222222
```

[21]:
```
y_predicted = model.predict(X_test)
```

[23]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_predicted)
cm
```

[23]:
```
array([[37,  0,  0,  0,  0,  0,  0,  0,  0,  0],
       [ 0, 27,  0,  0,  0,  0,  0,  0,  1,  0],
       [ 1,  0, 35,  0,  0,  0,  0,  0,  0,  0],
       [ 0,  0,  0, 31,  0,  0,  0,  0,  0,  1],
       [ 0,  0,  0,  0, 34,  0,  0,  0,  1,  0],
       [ 0,  0,  0,  0,  0, 42,  0,  0,  0,  0],
       [ 0,  1,  0,  0,  0,  0, 41,  0,  0,  0],
       [ 0,  0,  0,  0,  0,  0,  0, 41,  0,  0],
       [ 0,  2,  0,  0,  0,  1,  0,  1, 33,  0],
       [ 0,  1,  0,  0,  0,  0,  0,  0,  0, 29]], dtype=int64)
```

[25]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(cm, annot = True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

[25]:
```
Text(95.72222222222221, 0.5, 'Truth')
```

[ ]:
```
```