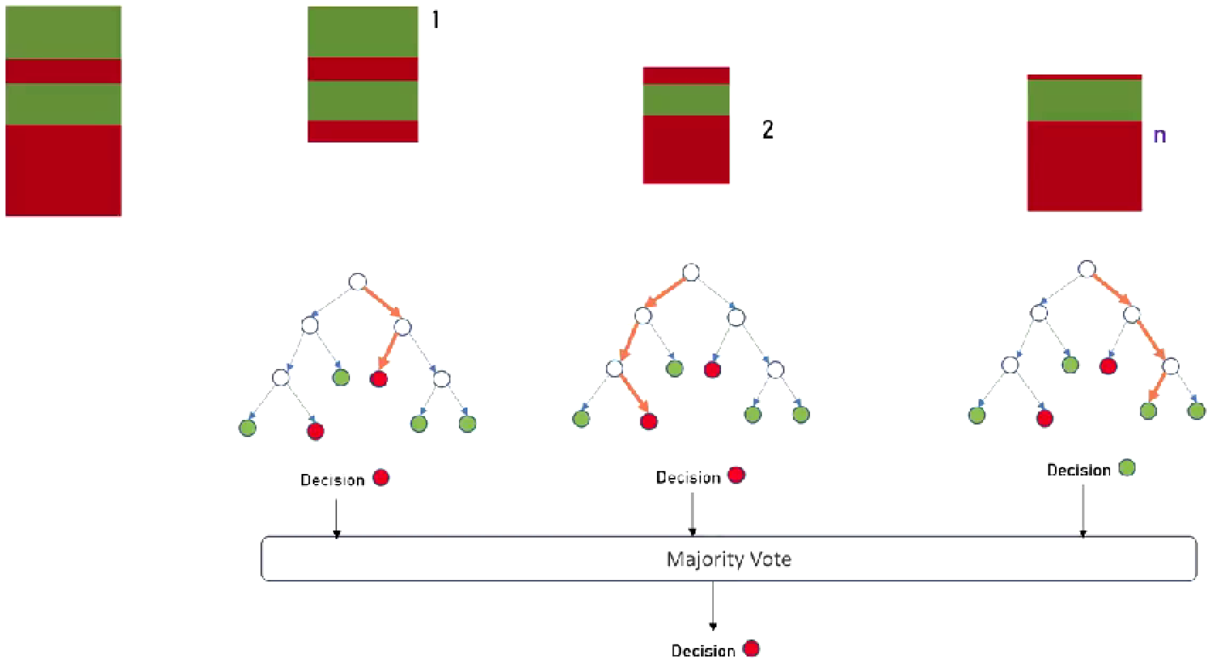
Random Forest Python Tutorial

What is Random Forset Algorithm?

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.



Advantage of Random Forest vs. Decision Tree:

With that said, random forests are a strong modeling technique and much more robust than a single decision tree. They aggregate many decision trees to limit overfitting as well as error due to bias and therefore yield useful results.

Problem

We are going to use Digits dataset from sklearn to make classification using Random Forest

In [2]:
```
import pandas as pd
from sklearn.datasets import load_digits
digits = load_digits()
```

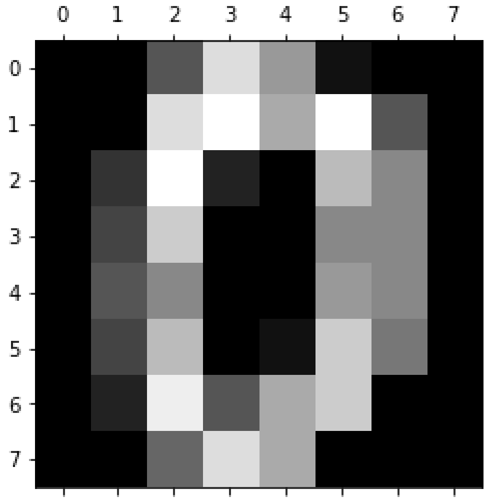
In [3]:
```
dir(digits)
```

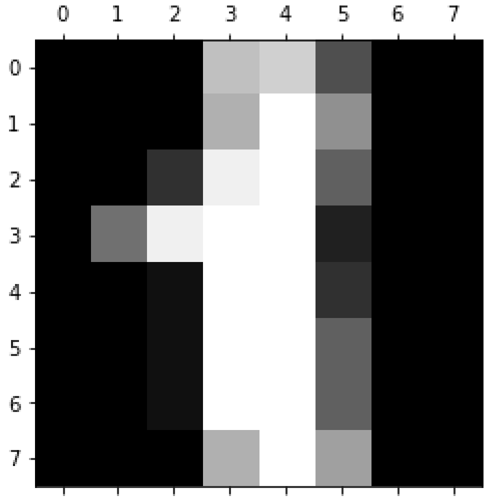
Out[3]:
```
['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_names']
```

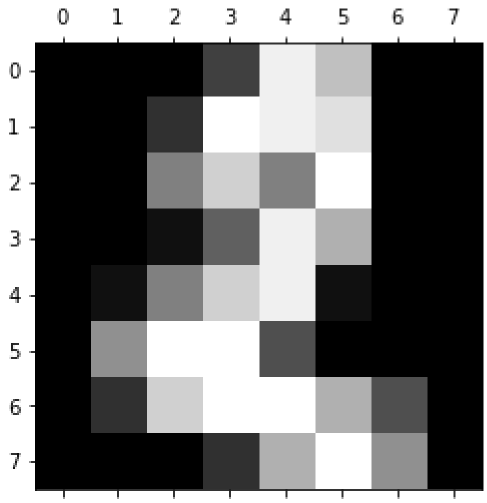
In [4]:
```
%matplotlib inline
import matplotlib.pyplot as plt
```

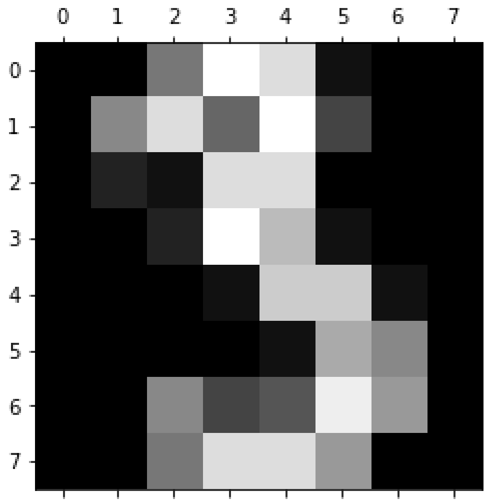
In [5]:
```
plt.gray()
for i in range(4):
    plt.matshow(digits.images[i])
```

```
<Figure size 432x288 with 0 Axes>
```









In [6]:

```
df = pd.DataFrame(digits.data)
df.head()
```

Out[6]:

```
     0    1    2     3     4     5    6    7    8    9  ...   54   55   56   57   58    59    60    61   62
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  0.0  6.0  13.0  10.0   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  0.0  0.0  11.0  16.0  10.0  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  0.0  0.0   3.0  11.0  16.0  9.0
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  0.0  7.0  13.0  13.0   9.0  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  0.0  0.0   2.0  16.0   4.0  0.0

5 rows × 64 columns
```

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

Out[7]:
```
array([0, 1, 2, ..., 8, 9, 8])
```

In [9]:
```
# Create new column in pandas DataFrame
df['target'] = digits.target
df.head()
```

Out[9]:

	0	1	2	3	4	5	6	7	8	9	...	55	56	57	58	59	60	61	62	63
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	6.0	13.0	10.0	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	0.0	11.0	16.0	10.0	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	0.0	3.0	11.0	16.0	9.0	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	7.0	13.0	13.0	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	0.0	2.0	16.0	4.0	0.0	0.0

5 rows × 65 columns

Train and the model and prediction

In [10]:
```
X = df.drop('target',axis='columns')
y = df.target
```

In [11]:
```
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)
```

In [12]:
```
len(X_train)
```

Out[12]:
```
1437
```

In [13]:
```
len(X_test)
```

Out[13]:
```
360
```

Use Random Forest Classifier to train the model

In [24]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

Out[24]:
```
RandomForestClassifier()
```

ensemble is used when you are using multiple algorithms to predict the outcome.

n_estimators by default is 100 i.e. it used 100 random trees. You can change it (n_estimators=200).

If the number of estimators increased the accuracy increased.

```
In [25]: model.score(X_test, y_test)
```

```
Out[25]: 0.9722222222222222
```

```
In [27]: y_predicted = model.predict(X_test)
```

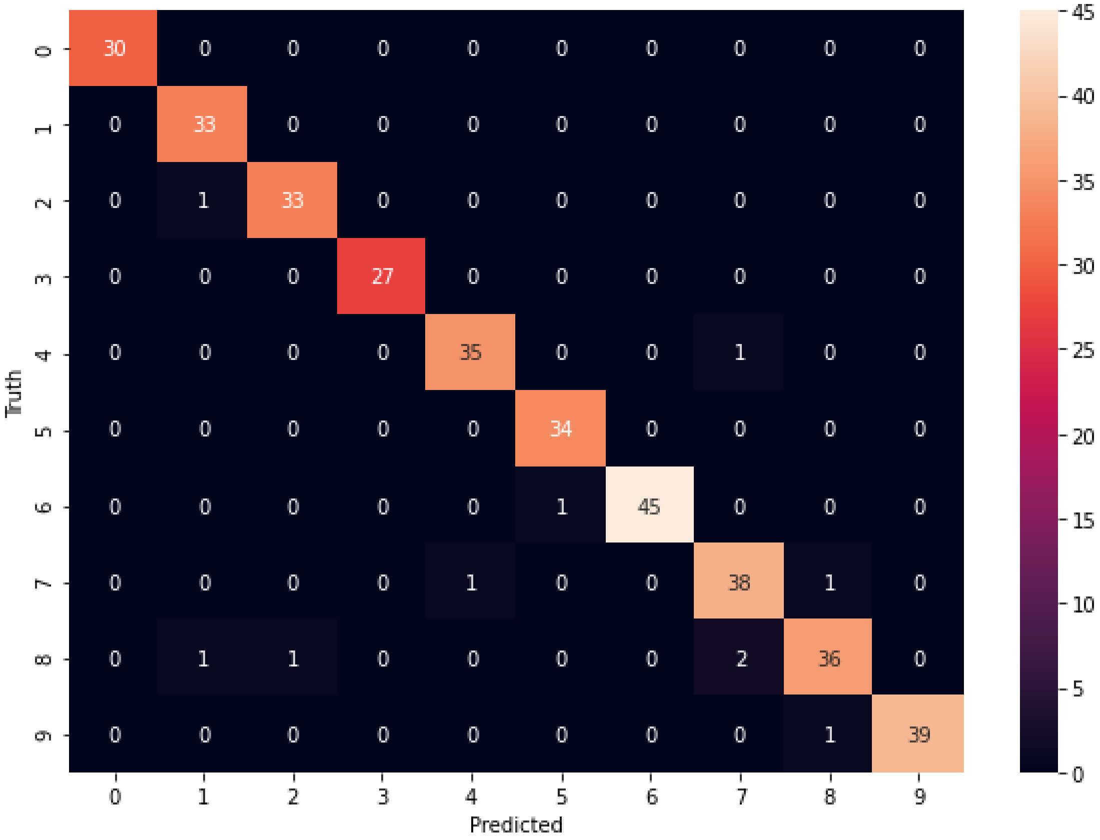
Confusion Matrix

```
In [28]: from sklearn.metrics import confusion_matrix
         cm = confusion_matrix(y_test, y_predicted)
         cm
```

```
Out[28]: array([[30,  0,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0, 33,  0,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  1, 33,  0,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0, 27,  0,  0,  0,  0,  0,  0],
                [ 0,  0,  0,  0, 35,  0,  0,  1,  0,  0],
                [ 0,  0,  0,  0,  0, 34,  0,  0,  0,  0],
                [ 0,  0,  0,  0,  0,  1, 45,  0,  0,  0],
                [ 0,  0,  0,  0,  1,  0,  0, 38,  1,  0],
                [ 0,  1,  1,  0,  0,  0,  0,  2, 36,  0],
                [ 0,  0,  0,  0,  0,  0,  0,  0,  1, 39]], dtype=int64)
```

In [29]:
```
%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')
```

Out[29]:
```
Text(69.0, 0.5, 'Truth')
```



Date	Author
2021-09-19	Ehsan Zia