Assignment 4

Problem definition:

Develop ML model for recognizing hand written digits with two classifier algorithm on publicaly available dataset with scikit-learn

Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible

I have imported matplotlib as plt and also used some libraries like numpy, pandas, etc.

Scikit-learn

Scikit-learn is an open source data analysis library, and the gold standard for Machine Learning (ML) in the Python ecosystem. Key concepts and features include: Algorithmic decision-making methods, including: Classification: identifying and categorizing data based on patterns

In alogorithm I used scikit learn to import the datasets

Datasets model

A dataset contains the source image or text data. A model is created after a dataset is trained. The model is the construct that returns

Seaborn

Seaborn is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions.

I have used seaborn as sns in the alogorithm

Svm

SVMs are used in applications like handwriting recognition, intrusion detection, face detection, email classification, gene classification, and in web pages. This is one of the reasons we use SVMs in machine learning. It can handle both classification and regression on linear and non-linear data

Plt

plt. show() starts an event loop, looks for all currently active figure objects, and opens one or more interactive windows that display your figure or figures.

Subplot

subplot is a side story that runs parallel to the main plot. It has a secondary strand of characters and events that can infuse important information into the main storyline.

Imshow

imshow(BW) displays the binary image BW in a figure. For binary images, imshow displays pixels with the value 0 (zero) as black and 1 as white

using plt , the subplot and imshow tis to deaclare the predicted variables

SVC(Super vector machine)

SVC, or Support Vector Classifier, is a supervised machine learning algorithm typically used for classification tasks. SVC works by mapping data points to a high-dimensional space and then finding the optimal hyperplane that divides the data into two classes

The two matrices that I have used in this algorithm are:

1. **accuracy_score**
2. **confusion_matrix**

Accuracy_score

Accuracy classification score. In multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must exactly match the corresponding set of labels in y_true.

Confusion_matrix

It is a table that is used in classification problems to assess where errors in the model were made. The rows represent the actual classes the outcomes should have been. While the columns represent the predictions we have made.

classifier

a classifier is a type of machine learning algorithm used to assign a class label to a data input. An example is an image recognition classifier to label an image (e.g., “car,” “truck,” or “person”).

The two classifier used in this algorithm are:

1. Decision tree classifier

2. Random forest classifier

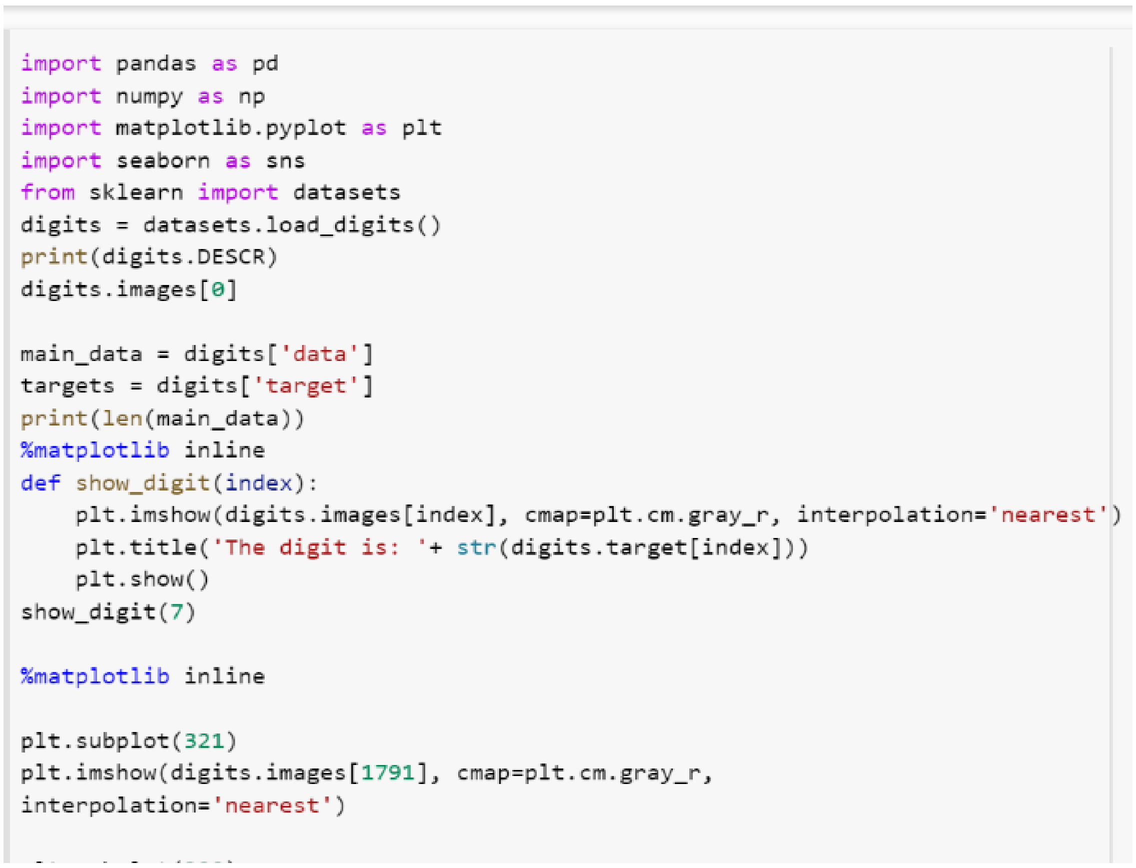
Decision tree classifier

- The decision tree classifier (Pang-Ning et al., 2006) creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute.
- Decision trees are used for handling non-linear data sets effectively

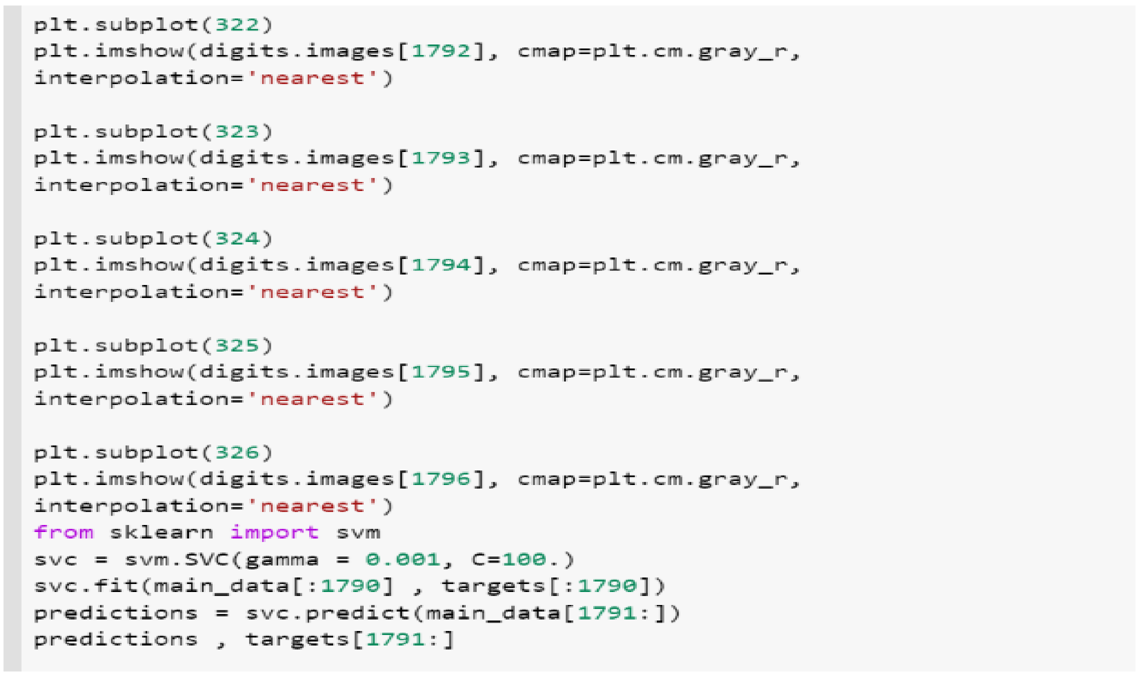
Random forest classification

- A random forest classifier. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
- A random forest—as the name suggests—consists of multiple decision trees each of which outputs a prediction. When performing a classification task, each decision tree in the random forest votes for one of the classes to which the input belongs

Input :



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import datasets
digits = datasets.load_digits()
print(digits.DESCR)
digits.images[0]

main_data = digits['data']
targets = digits['target']
print(len(main_data))
%matplotlib inline
def show_digit(index):
    plt.imshow(digits.images[index], cmap=plt.cm.gray_r, interpolation='nearest')
    plt.title('The digit is: '+ str(digits.target[index]))
    plt.show()
show_digit(7)

%matplotlib inline

plt.subplot(321)
plt.imshow(digits.images[1791], cmap=plt.cm.gray_r,
interpolation='nearest')
```



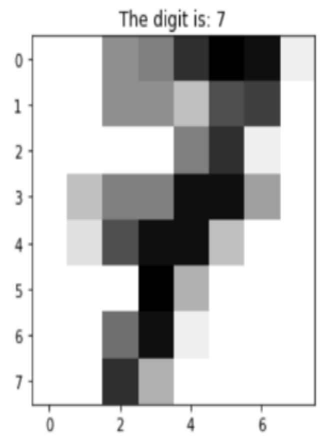
```
plt.subplot(322)
plt.imshow(digits.images[1792], cmap=plt.cm.gray_r,
interpolation='nearest')

plt.subplot(323)
plt.imshow(digits.images[1793], cmap=plt.cm.gray_r,
interpolation='nearest')

plt.subplot(324)
plt.imshow(digits.images[1794], cmap=plt.cm.gray_r,
interpolation='nearest')

plt.subplot(325)
plt.imshow(digits.images[1795], cmap=plt.cm.gray_r,
interpolation='nearest')

plt.subplot(326)
plt.imshow(digits.images[1796], cmap=plt.cm.gray_r,
interpolation='nearest')
from sklearn import svm
svc = svm.SVC(gamma = 0.001, C=100.)
svc.fit(main_data[:1790] , targets[:1790])
predictions = svc.predict(main_data[1791:])
predictions , targets[1791:]
```

```python
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier(criterion = 'gini')
dt.fit(main_data[:1600] , targets[:1600])
prediction2 = dt.predict(main_data[1601:])

from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
confusion_matrix(targets[1601:], prediction2)
accuracy_score(targets[1601:], prediction2)

from sklearn.ensemble import RandomForestClassifier
rc = RandomForestClassifier(n_estimators = 150)
rc.fit(main_data[:1500], targets[:1500])
prediction3= rc.predict(main_data[1501:])
accuracy_score(targets[1501:], prediction3)
```

Output:

```
.. _digits_dataset:

Optical recognition of handwritten digits dataset
--------------------------------------------------

**Data Set Characteristics:**

    :Number of Instances: 1797
    :Number of Attributes: 64
    :Attribute Information: 8x8 image of integer pixels in the range 0..16.
    :Missing Attribute Values: None
    :Creator: E. Alpaydin (alpaydin '@' boun.edu.tr)
    :Date: July; 1998

This is a copy of the test set of the UCI ML hand-written digits datasets
https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

The data set contains images of hand-written digits: 10 classes where
each class refers to a digit.

Preprocessing programs made available by NIST were used to extract
normalized bitmaps of handwritten digits from a preprinted form. From a
total of 43 people, 30 contributed to the training set and different 13
to the test set. 32x32 bitmaps are divided into nonoverlapping blocks of
4x4 and the number of on pixels are counted in each block. This generates
an input matrix of 8x8 where each element is an integer in the range
0..16. This reduces dimensionality and gives invariance to small
distortions.
```

```
1797
```



```
0.9256756756756757
```