Logistic Regression

April 4, 2022

0.1 Logistic Regression for Classification - target takes specific values only

[1]:
```
from sklearn.datasets import load_digits
```

[2]:
```
digits = load_digits()
```

[3]:
```
print(digits.DESCR)
```

```
.. _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+Digit
s

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.
```

```
For info on NIST preprocessing routines, see M. D. Garris, J. L. Blue, G.
T. Candela, D. L. Dimmick, J. Geist, P. J. Grother, S. A. Janet, and C.
L. Wilson, NIST Form-Based Handprint Recognition System, NISTIR 5469,
1994.

.. topic:: References

  - C. Kaynak (1995) Methods of Combining Multiple Classifiers and Their
    Applications to Handwritten Digit Recognition, MSc Thesis, Institute of
    Graduate Studies in Science and Engineering, Bogazici University.
  - E. Alpaydin, C. Kaynak (1998) Cascading Classifiers, Kybernetika.
  - Ken Tang and Ponnuthurai N. Suganthan and Xi Yao and A. Kai Qin.
    Linear dimensionalityreduction using relevance weighted LDA. School of
    Electrical and Electronic Engineering Nanyang Technological University.
    2005.
  - Claudio Gentile. A New Approximate Maximal Margin Classification
    Algorithm. NIPS. 2000.
```

[4]:
```python
import matplotlib.pyplot as plt
```

[5]:
```python
#shows 2d data, each row contains multiple columns with meaning
digits.data
```

[5]:
```
array([[ 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.]])
```

[6]:
```python
#dimension of actual data
digits.data.shape
```

[6]:
```
(1797, 64)
```

[7]:
```python
d=digits.data[0:500]
```

[8]:
```python
d.shape
```

[8]:
```
(500, 64)
```

[9]:
```python
#target or label dimension
digits.target.shape
```

[9]:
```
(1797,)
```

```
[14]: #taking only the first image pixel data
      image = digits.data[0]
```

```
[15]: #image is represented as collection of 64 pixel values
      print(image)
```

```
[ 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.]
```

```
[16]: #determine class of the 64 pixels stored in image
      digits.target[0]
```

```
[16]: 0
```

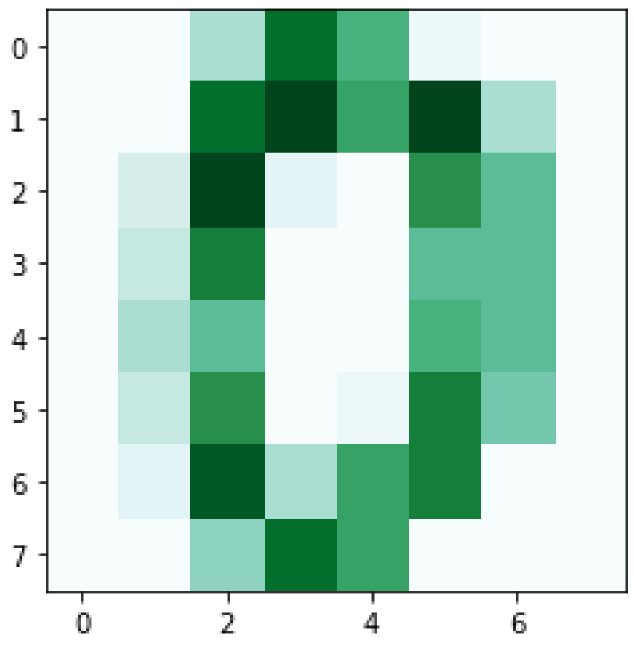
```
[17]: import numpy as np
```

```
[18]: np.reshape(image,(8,8))
```

```
[18]: array([[ 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.]])
```

```
[19]: plt.imshow(np.reshape(image, (8,8)), cmap='BuGn')
```

```
[19]: <matplotlib.image.AxesImage at 0x7f8e5d16ccd0>
```



```
[20]: from sklearn.model_selection import train_test_split
```

```
[21]: #split the dataset using train_test_split function, pass train data, labels,␣
      ↪and test data ratio
      x_train, x_test, y_train, y_test =
      train_test_split(digits.data,digits.target,test_size=0.2)
```

```
[22]: from sklearn.linear_model import LogisticRegression
```

```
[23]: logReg1 = LogisticRegression(max_iter=10000)
```

```
[24]: logReg1.fit(x_train, y_train)
```

```
[24]: LogisticRegression(max_iter=10000)
```

```
[25]: y_pred=logReg1.predict(x_test)
```

```
[26]: logReg1.score(x_test,y_test)
```

```
[26]: 0.9611111111111111
```

```
[ ]:
```