scikit-datasets

2018 年 6 月 22 日

1 scikit-learn 內建資料集

1.1 介紹

scikit 內建一些供我們可以練習基礎的資料集，你可以先藉由這些內建資料集來確定你的模型沒問題以後，再應用到現實世界的資料集機器學習的兩大類問題，我們可以分別先使用裡面的兩個資料集來試試看 1. 分類問題: 目標不是連續的數值，而是標籤類型的，我們要預判這些標籤 -> 鳶尾花數據集 2. 迴歸問題: 目標是連續的數值，我們要預測這些連續的數值 -> 波士頓房產數據集

1.2 使用函式庫

請使用命令列或者 pycharm 安裝以下四個函式庫

1. pandas: 在處理表格和矩陣的時候，我還是習慣統一使用 pandas 做為我們的處理工具
2. numpy: 提供許多數學運算，包括許多的矩陣運算
3. scipy: 提供許多工程運算，包括線性代數，微積分等等
4. scikit-learn: 實作了許多機器學習基本方法的一個函式庫，也提供我們很多處理資料的方法以及內建資料集，基於 scipy 和 numpy

1.3 官方網站和例子

1. 官方網站: http://scikit-learn.org/stable/index.html
2. 內建資料集: http://scikit-learn.org/stable/modules/classes.html#module-sklearn.datasets

1.4 鳶尾花資料集 (分類問題)

共有三種不同的鳶尾花 (圖片引用自 wiki 百科)，我們要用花瓣 (petal) 的長寬和花萼 (sepal) 的長寬來做出分類

1. 山鳶尾 (Setosa) - 資料集裡以 0 做表示
2. 變色鳶尾 (Versicolor) - 資料集裡以 1 做表示
3. 維吉尼亞鳶尾 (Virginica) - 資料集裡以 2 做表示

```
In [1]: import pandas as pd
        # 為了顯示的漂亮, 我刻意的把印出來的 row 只顯示 10 個和 column 只顯示十個
        # 大家練習的時候可以去掉下面兩行
        pd.set_option('display.max_rows', 10)
        pd.set_option('display.max_columns', 10)

In [2]: from sklearn.datasets import load_iris

In [3]: iris = load_iris()
        # 讀者可以自行把下面行的註解拿掉 印出 iris
        # iris

In [4]: # 先用 data 和他的 column 創出 df
        df = pd.DataFrame(data= iris['data'], columns= iris['feature_names'])
        df["target"] = iris["target"]
        df
```

Out[4]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0
..	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

```
3         0
4         0
..      ...
145       2
146       2
147       2
148       2
149       2

[150 rows x 5 columns]
```

1.5 波士頓地產資料集 (迴歸問題)

把波士頓地區的房價和當地區的特徵列出來，由於房價是一個連續的值，所以我們通常會拿來學習回歸 (以下翻譯引用自 http://sklearn.apachecn.org/cn/0.19.0/sklearn/datasets/descr/boston_house_prices.html)

```
In [5]: from sklearn.datasets import load_boston
        boston = load_boston()
        # 讀者可以自行把下面行的註解拿掉 印出 boston
        # boston
```

```
In [6]: df = pd.DataFrame(data = boston['data'], columns = boston['feature_names'])
        df['target'] = boston['target']
        df
```

```
Out[6]:         CRIM    ZN  INDUS  CHAS    NOX   ...     TAX  PTRATIO       B  LSTAT  \
        0    0.00632  18.0   2.31   0.0  0.538   ...   296.0     15.3  396.90   4.98
        1    0.02731   0.0   7.07   0.0  0.469   ...   242.0     17.8  396.90   9.14
        2    0.02729   0.0   7.07   0.0  0.469   ...   242.0     17.8  392.83   4.03
        3    0.03237   0.0   2.18   0.0  0.458   ...   222.0     18.7  394.63   2.94
        4    0.06905   0.0   2.18   0.0  0.458   ...   222.0     18.7  396.90   5.33
        ..       ...   ...    ...   ...    ...   ...     ...      ...     ...    ...
        501  0.06263   0.0  11.93   0.0  0.573   ...   273.0     21.0  391.99   9.67
        502  0.04527   0.0  11.93   0.0  0.573   ...   273.0     21.0  396.90   9.08
        503  0.06076   0.0  11.93   0.0  0.573   ...   273.0     21.0  396.90   5.64
        504  0.10959   0.0  11.93   0.0  0.573   ...   273.0     21.0  393.45   6.48
        505  0.04741   0.0  11.93   0.0  0.573   ...   273.0     21.0  396.90   7.88
```

```
     target
0      24.0
1      21.6
2      34.7
3      33.4
4      36.2
..      ...
501    22.4
502    20.6
503    23.9
504    22.0
505    11.9

[506 rows x 14 columns]
```