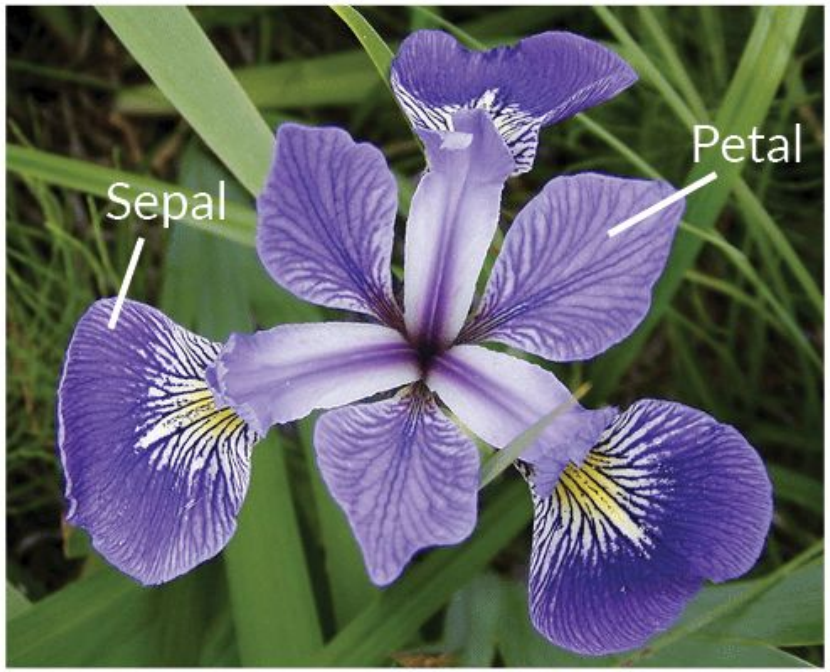
LAB3-Logistic regression

國立雲林科技大學
夏世昌 特聘教授 / 電子工程系
王斯弘 助理教授 / 前瞻學位學士學程

2022/10/05

dataset_*Fisher's Iris*



Iris Versicolor

Iris Setosa

Iris Virginica

費雪鳶尾花卉資料集

花萼長度	花萼寬度	花瓣長度	花瓣寬度	屬種
5.1	3.5	1.4	0.2	*setosa*
4.9	3.0	1.4	0.2	*setosa*
4.7	3.2	1.3	0.2	*setosa*
⋮				
5.3	3.7	1.5	0.2	*setosa*
5.0	3.3	1.4	0.2	*setosa*
7.0	3.2	4.7	1.4	*versicolor*
6.4	3.2	4.5	1.5	*versicolor*
⋮				

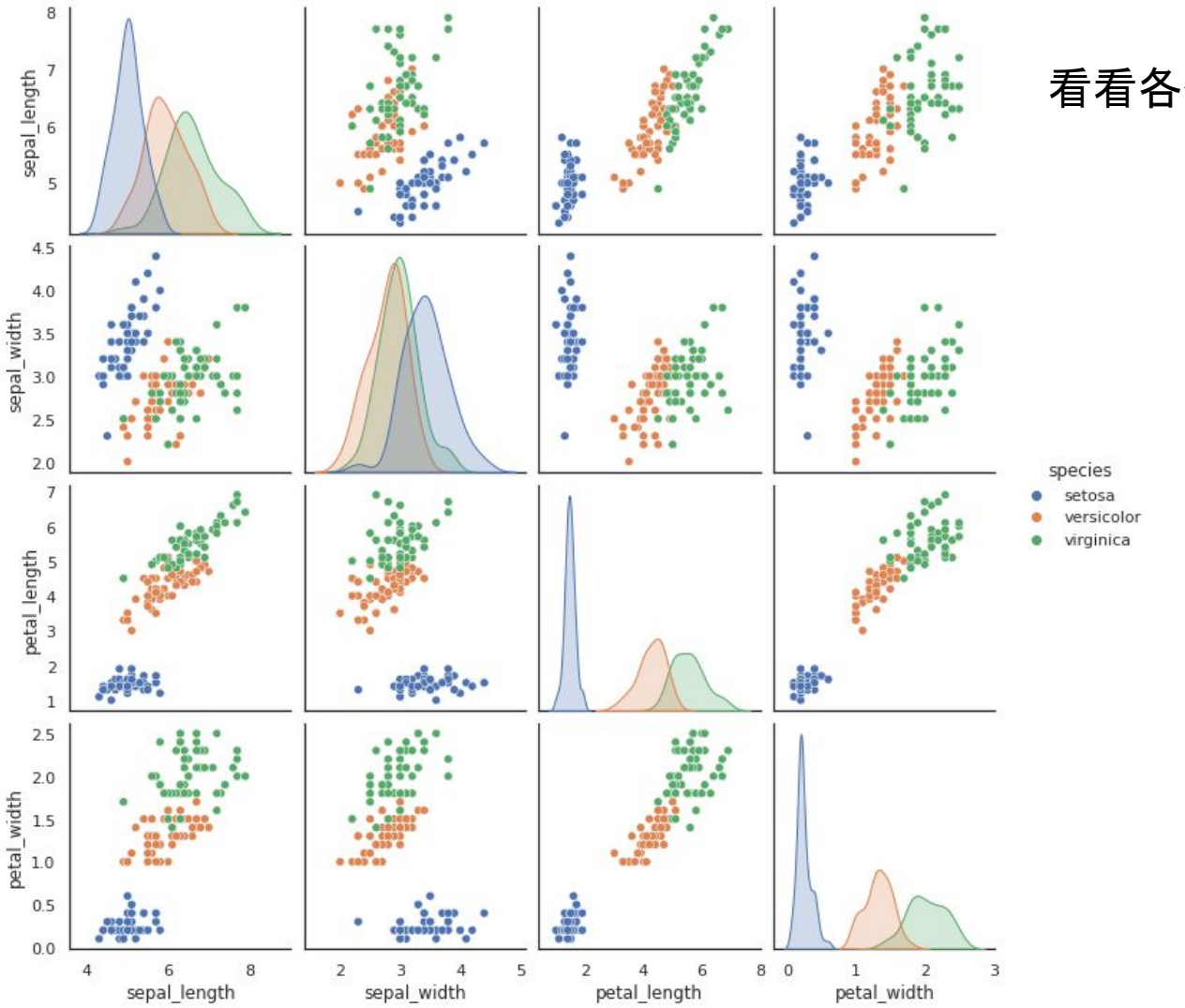
50筆

50筆

Sepal 花萼
petal 花瓣

每種屬性都有50筆資料

Preparation(前處理)



看看各個特徵的相關性(Correlation)

prepared code

```
import seaborn as sns

iris_sns = sns.load_dataset("iris")
print(iris_sns)
sns.pairplot(iris_sns, hue="species")
```

Seaborn 視覺化工具

```
pip install seaborn
```

seaborn為python繪圖函式庫，以matplotlib為基礎封裝了許多實用的統計圖表

https://pyecontech.com/2018/08/28/python_seaborn_installation_replot/

Import library

```
import matplotlib.pyplot as plt
import numpy as np
import math
import random
from sklearn import datasets
```

Import library

from sklearn import datasets

scikit-learn，又寫作**sklearn**，是一個開源的基於python語言的機器學習工具包。

http://scikit-learn.org/stable/index.html

Import library

安裝sklearn

& pip install -U scikit-learn

```
(keras 2) C:\Users\USER>pip install -U scikit-learn
```

Import library

Import math

math — Mathematical functions¶

This module provides access to the mathematical functions defined by the C standard.

https://docs.python.org/3/library/math.html

Import library

Import random

`random` — Generate pseudo-random numbers

This module implements pseudo-random number generators for various distributions.

https://docs.python.org/3/library/random.html

dataset

```
iris = datasets.load_iris() #載入sklearn中iris資料集
X = iris.data
Y = iris.target
```

資料處理

```
dataset = []
target_label = 0 # label_0設為目標
for index, x in enumerate(X):
    transform_label = None
    if Y[index] == target_label:
        transform_label = 1 # label0
    else:
        transform_label = 0 # label0以外
    x = [x[0], x[2]]#取X[0]和X[2]的資料
    dataset.append((x,transform_label))#結合x和label資料
dataset = np.array(dataset)
```

定義函式

```python
#sigmoid公式
def sigmoid(z):
    return 1 / (1 + np.exp(-z))
```

Sigmoid 函數（sigmoid function）

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^{x}}{e^{x}+1}.$$

dataset

Sigmoid() 函數簡單來講就是個映射函數，將任何變量映射到 [0, 1] 之間。
通常被用來當作機器學習領域 (Machine Learning) 神經網路的激活函數 (Activation Function)。

定義函式

```python
#計算平均梯度
def gradient(dataset, w):
    index = random.randint(0, len(dataset) - 1) #隨機取data
    x, y = dataset[index]
    x = np.array(x)
    error = sigmoid(w.T.dot(x))
    g = (error - y) * x
    return g
```

dot():矩陣相乘

```
sigmoid(w.T.dot(x))
```

T:矩陣轉置

定義函式

```python
#計算loss
def cost(dataset, w):
    total_cost = 0
    for x,y in dataset:
        x = np.array(x)
        error = sigmoid(w.T.dot(x))
        total_cost += abs(y - error)
    return total_cost
```

定義函式

abs()為python內建函式，可直接呼叫。

abs()函式可返回一個數的絕對值，()中引數可以是整數或浮點數，如果引數是複數，則返回這個複數的模

定義函式

```
#logistic_regression
def logistic_regression(dataset):
    w = np.zeros(2)      #權重初始化
    limit = 1500         #跑幾次
    eta = 0.0001             #初始學習率
    costs = []
    for i in range(limit):
        current_cost = cost(dataset, w)
        if i % 100 == 0:
            print ("epoch = " + str(i/100 + 1) + ": current_cost = ", current_cost,": w = ", w)
        costs.append(current_cost)
        w = w - eta * gradient(dataset, w)#更新權重
        eta = eta * 0.98 #衰減學習率
    plt.plot(range(limit), costs)  #畫出loss曲線
    plt.show()
    return w,(limit, costs)
```

執行主程式

```
def main():
    w = logistic_regression(dataset)
    #畫圖
    ps = [v[0] for v in dataset]
    label = [v[1] for v in dataset]

    fig = plt.figure()
    ax1 = fig.add_subplot(111)
    #plot via label
    tpx=[]
    for index, label_value in enumerate(label):
        px=ps[index][0]
        py=ps[index][1]
        tpx.append(px)
        if label_value == 1:
            ax1.scatter(px, py, c='b', marker="o", label='O') #label = 1 以 O 表示
        else:
            ax1.scatter(px, py, c='r', marker="x", label='X') #label = 1 以 X 表示

    l = np.linspace(min(tpx),max(tpx))
    a,b = (-w[0][0]/w[0][1], w[0][0])
    ax1.plot(l, a*l + b, 'g-')
    plt.show()

if __name__ == '__main__':
    main()                                  #執行main
```

執行主程式

```
fig = plt.figure()
ax1 = fig.add_subplot(111)
```

建立初始圖框

ax = fig.add_subplot(111)的意思是在fig圖裡切成1塊，亦可用(1,1,1)表示。

執行主程式

```
ax1 = fig.add_subplot(3,4,10)
```

ax = fig.add_subplot(3,4,10) 的意思是:在fig圖裡分割成3行4列, 影象畫在從左到右從上到下的第10塊

執行主程式

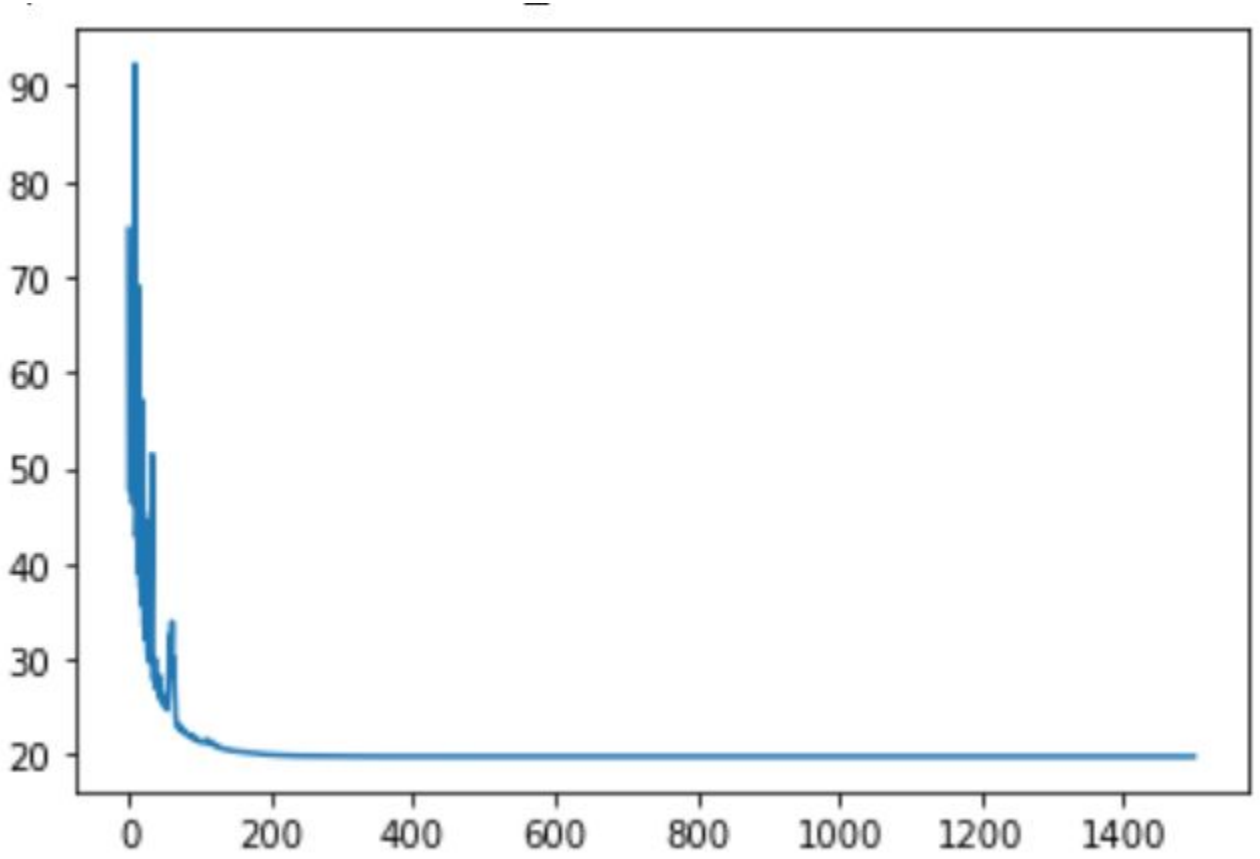
```
if label_value == 1:
    ax1.scatter(px, py, c='b', marker='o', label='O')
else:
    ax1.scatter(px, py, c='r', marker='x', label='X')
```

c	顏色
marker	圖示
label	標籤

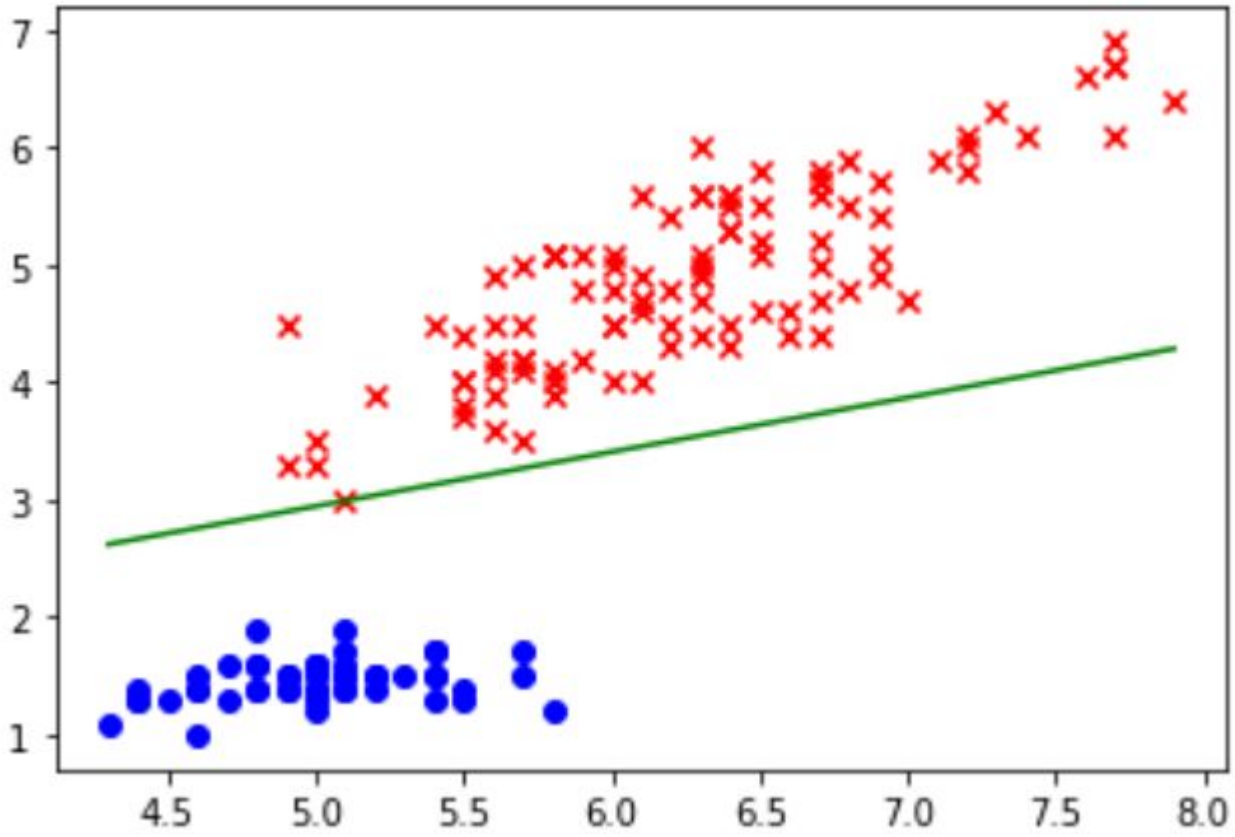
成果

```
epoch = 1.0: current_cost =  75.0 : w =  [0. 0.]
epoch = 2.0: current_cost =  21.416408250117353 : w =  [ 0.56235226 -1.29423644]
epoch = 3.0: current_cost =  20.01806236745835 : w =  [ 0.61987343 -1.35456694]
epoch = 4.0: current_cost =  19.854118996084743 : w =  [ 0.62910898 -1.36181737]
epoch = 5.0: current_cost =  19.83384173160945 : w =  [ 0.63078948 -1.3626297 ]
epoch = 6.0: current_cost =  19.83052932664373 : w =  [ 0.63077205 -1.36282826]
epoch = 7.0: current_cost =  19.830063410616617 : w =  [ 0.63076144 -1.36285802]
epoch = 8.0: current_cost =  19.830000402600394 : w =  [ 0.63075981 -1.36286209]
epoch = 9.0: current_cost =  19.829994251561473 : w =  [ 0.63076013 -1.36286238]
epoch = 10.0: current_cost =  19.829993502038374 : w =  [ 0.6307602  -1.36286241]
epoch = 11.0: current_cost =  19.829993393309557 : w =  [ 0.63076021 -1.36286242]
epoch = 12.0: current_cost =  19.829993379742934 : w =  [ 0.63076021 -1.36286242]
epoch = 13.0: current_cost =  19.82999337768183 : w =  [ 0.63076021 -1.36286242]
epoch = 14.0: current_cost =  19.829993377418152 : w =  [ 0.63076021 -1.36286242]
epoch = 15.0: current_cost =  19.82999337737491 : w =  [ 0.63076021 -1.36286242]
```

成果



成果



總結

1.知道gradient、loss、sigmoid -> 調參數

2.看懂程式

----------以下進修----------

1.能利用其他特徵去分類種類

2.探究資料的關係

END