**類神經網路作業一:**

**Perceptron與logistic regression實作**

**說明文件**

**學號:10427238**

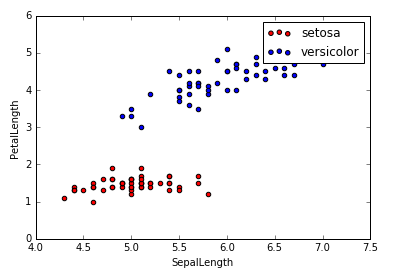
**資訊四乙 陳冠友**

* **資料前處理**

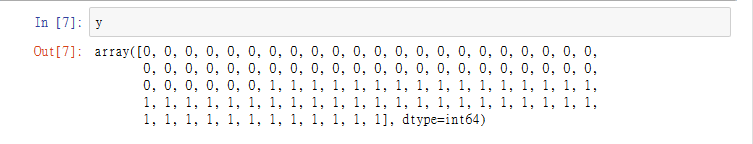
這次兩個分類器的實作都是使用iris dataset，並且只取出其中兩類(setosa, versicolor)。且為了資料視覺化的方便，只取其中兩個特徵(SepalLength, PetalLength)。

在Perceptron中，我將setosa與versicolor分別標記為class 1與class -1。而在logistic regression中，則將setosa與versicolor分別標記為class 0與class 1。

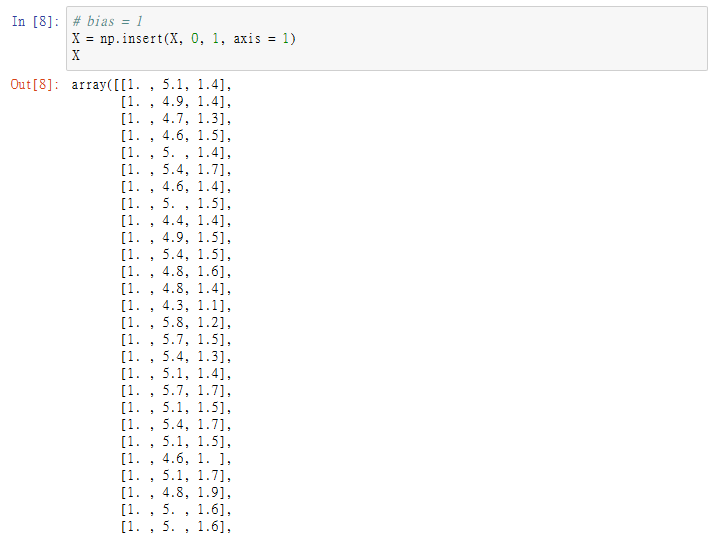
而training data中均為3 dimensional vector，且第一維均設定為1。



**(圖1):setosa與versicolor的分布**



**(圖2):預期輸出值(以logistic regression為例)**

****

**(圖三):training dataset格式 [x0, SepalLength, PetalLength]**

* **Perceptron程式說明**

Perceptron(X, y, l\_rate):

此function內部實作Perceptron演算法，且必須傳入三個參數，分別是training data、label data以及learning rate。而weight vector初始值為[-1,1,1]。

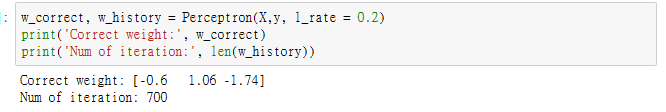
function內部除了實作Perceptron演算法之外，也在內部定義了sgn() function，其數學定義如下:

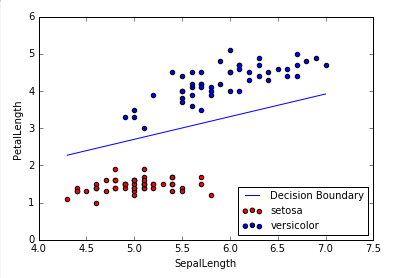
此function作為Perceptron的activation function。

而Perceptron演算法的實作理念為，先計算權重向量與training data的內積(即)，在將此內積值透過activation function: sgn()得到輸出。最後將輸出值與label data比較，若分類錯誤就進行修正。

而在每一輪的運算中，都會紀錄上一輪得到的權重以及這一輪更新的新權重。當舊的權重向量與新的權重向量之距離<0.1 (即舊的decision boundary與新的decision boundary其法向量距離小於0.1)，就代表訓練完成，即可結束訓練。

最後會回傳所得到的正確權重值以及訓練次數。

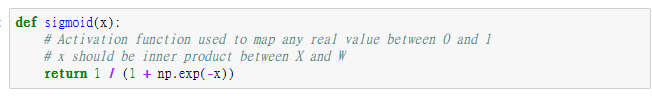




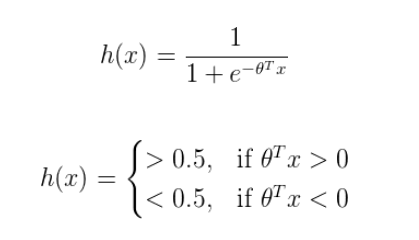
**(圖四):當learning rate為0.2時，所求得的decision boundary與weight**

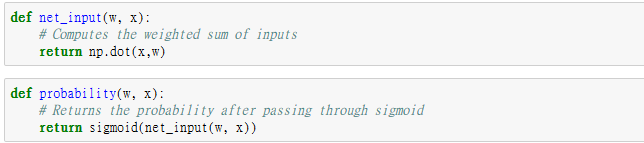
* **Logistic Regression程式說明**

首先實作sigmoid function:

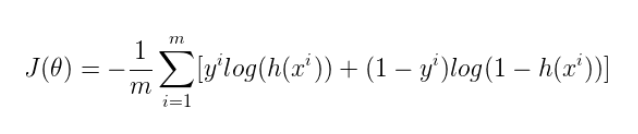


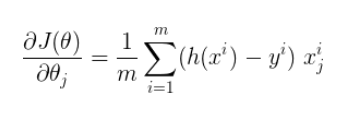
接著實作logistic regression的hypothesis:

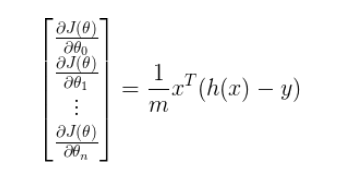


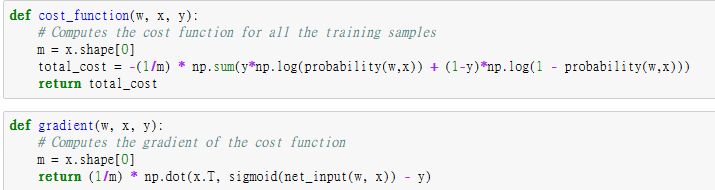


最後實作logistic regression的cost function以及gradient的計算:









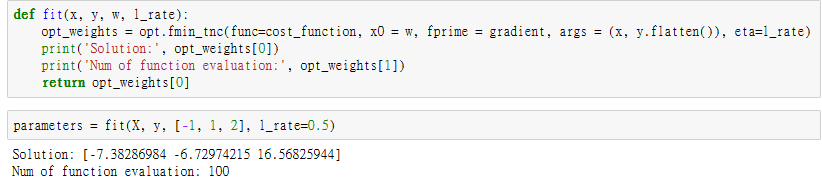
最後實作fit() function。使用者只需透過此function並且傳入training data, label data, weight vector以及learning rate，該function將會回傳算出的optimal weight vector。

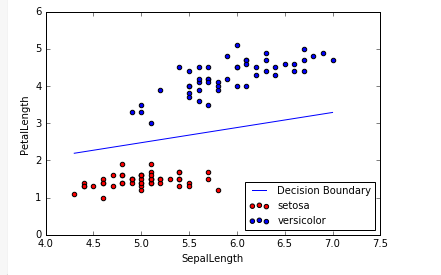
這個function透過Scipy library中的fmin\_tnc() function，

此function可幫助我們求cost function的最小值，只需要給予cost function的定義、gradient如何計算以及data sample即可。

下圖為fit function的實作以及實驗結果。

(weight vector:[-1,1,2] , learning rate:0.5)





**(Resources:**[**Building a logistic regression in python**](%20https:/towardsdatascience.com/building-a-logistic-regression-in-python-301d27367c24)**)**