1. **程式簡介**

目前程式只針對二維兩群的資料做分類，以HTML、CSS做介面設計，搭配JavaScript來撰寫多層感知器(MLP)，未使用JQuery等其他library的語法。MLP架構只有兩層，第一層隱藏層有兩個神經元，第二層輸出層只有一個神經元，如圖1所示，每個神經元的活化函數都採用Sigmoid function。在繪製資料圖案的部分，資料會先做正規化，目的是為了讓資料可以均勻分布在畫布上，因此每個資料集對應的畫布座標軸數值會不一樣，如圖2所示。

output

**圖1**

資料集x最小值

資料集x最大值

資料集y最小值

資料集y最大值

**畫布**

**圖2**

1. **程式執行說明**

直接打開html檔(建議用Chrome開啟)，選取要匯入的資料集，匯入資料集時會在右方繪製原始資料分類，並且初始化鍵結值，在下方顯示目前鍵結值的辨識率。接著設定學習率和學習次數，也可以另外勾選辨識率作為收斂條件，然後按「開始訓練」按鈕即可訓練。訓練完成後會跳出對話框提醒，下方會顯示訓練結果(包含鍵結值、訓練資料辨識率、測試資料辨識率、RMSE)，也會把資料圖像化繪製出來。另外會檢查輸入數值的範圍(學習率[0,1]、辨識率(1,100))，若超出範圍會有提示訊息提醒使用者。學習率預設值是0.05，訓練次數預設值是500，辨識率預設值90%，但預設不會勾選。

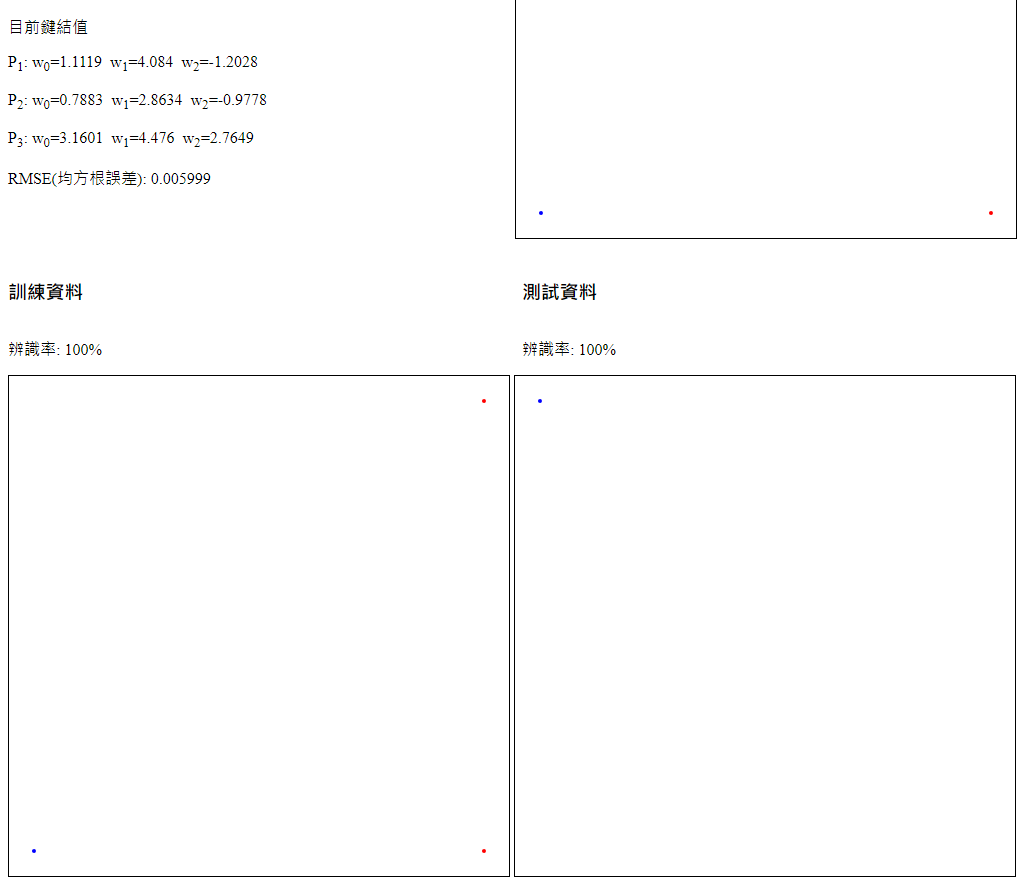
1. **實驗結果**

* **perceptron1.txt**

**初始畫面**

****

**訓練結果**

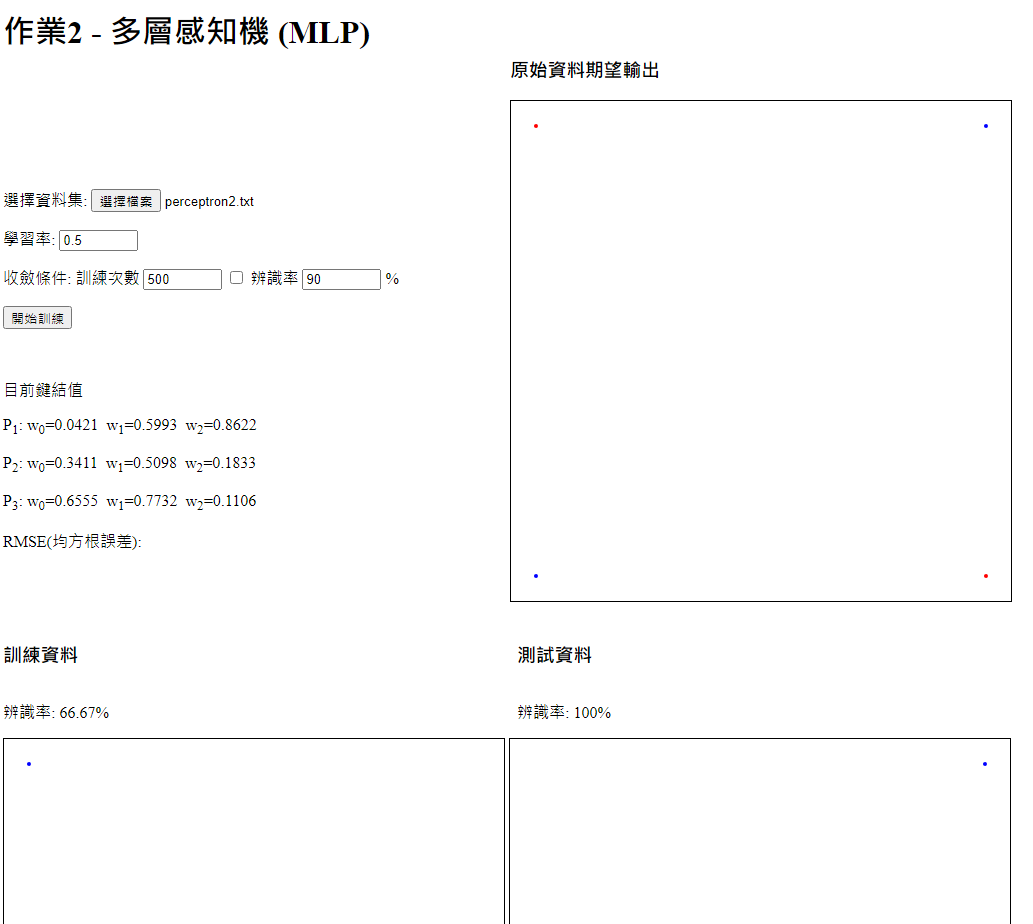
****

**說明**

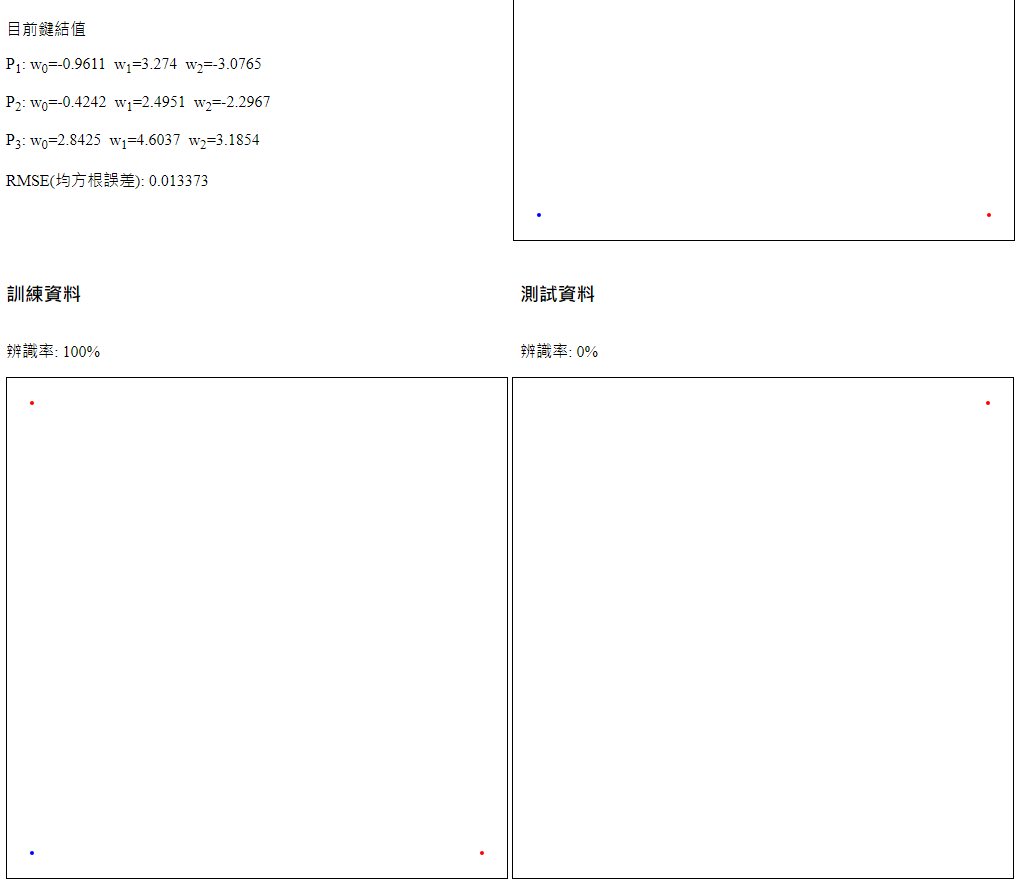
用預設值訓練會因為學習率太小，學不到100%的辨識率，因此將學習率設定為0.5，便能達到100%辨識率。

* **perceptron2.txt**

**初始畫面**



**訓練結果**

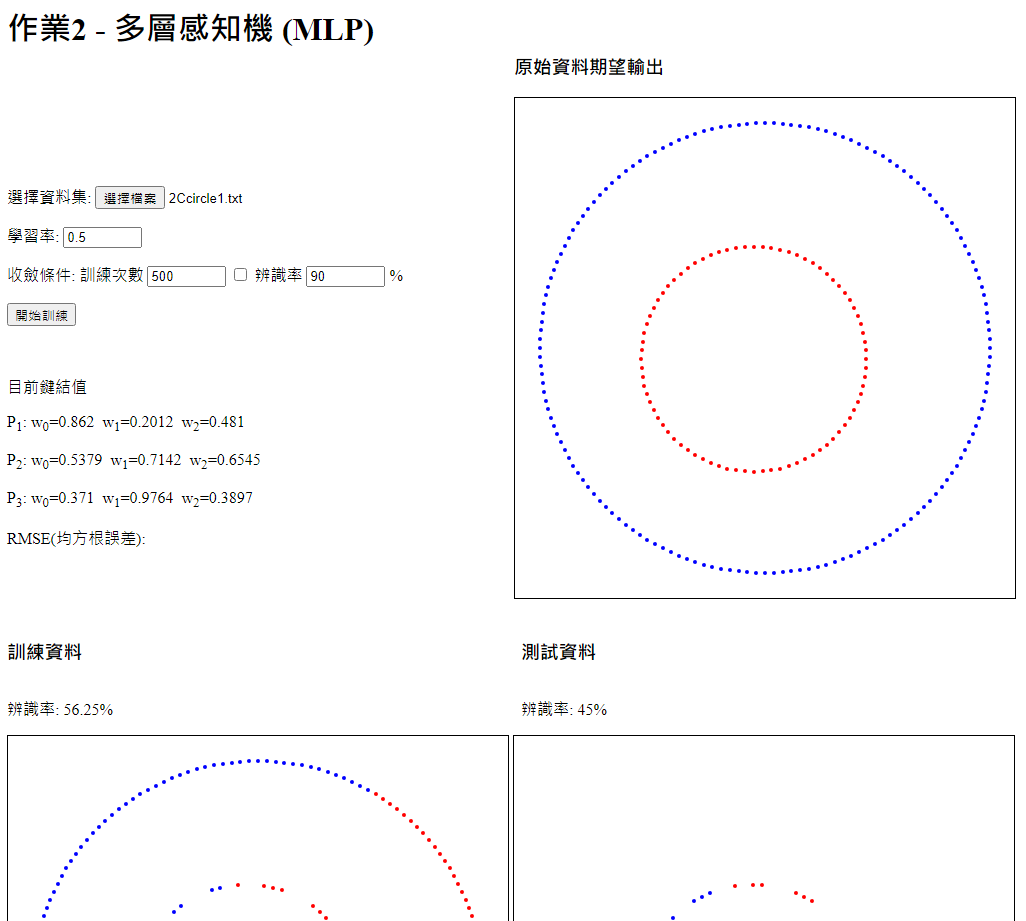


**說明**

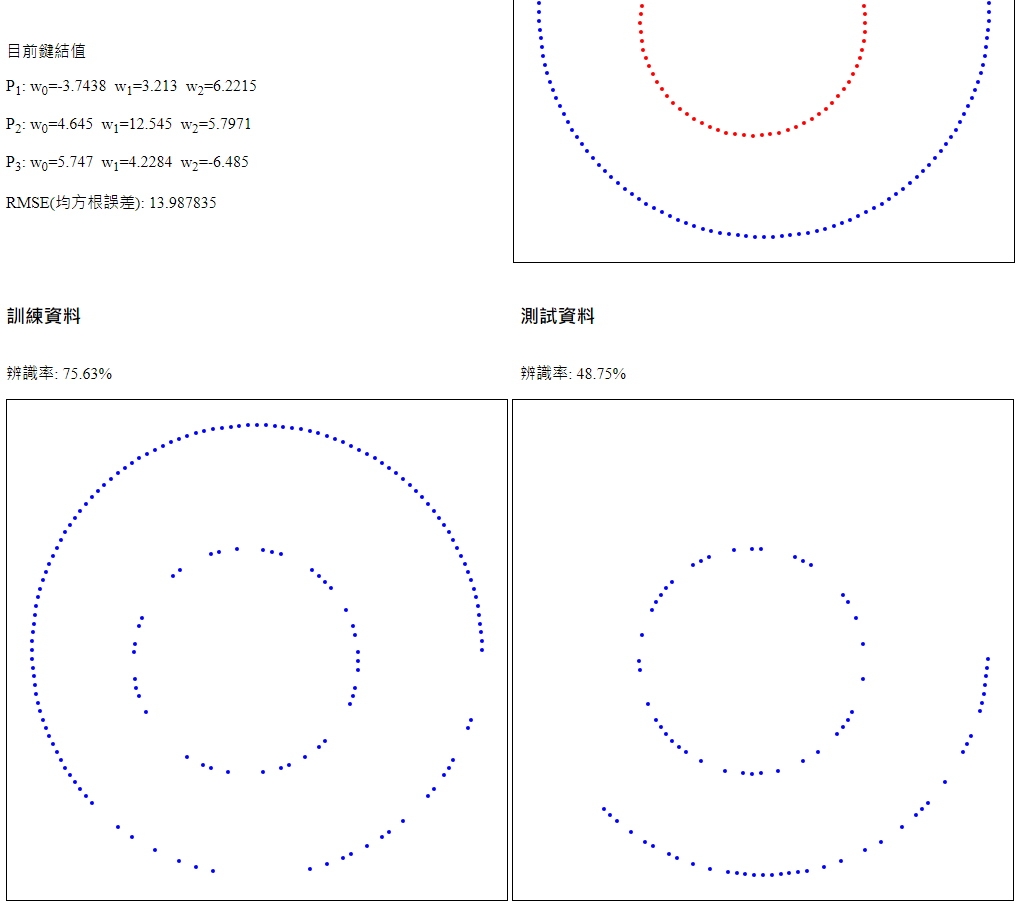
一樣只調整學習率，設為0.5，訓練資料集達到100%辨識率，而測試資料集因為沒學習到所以0%。

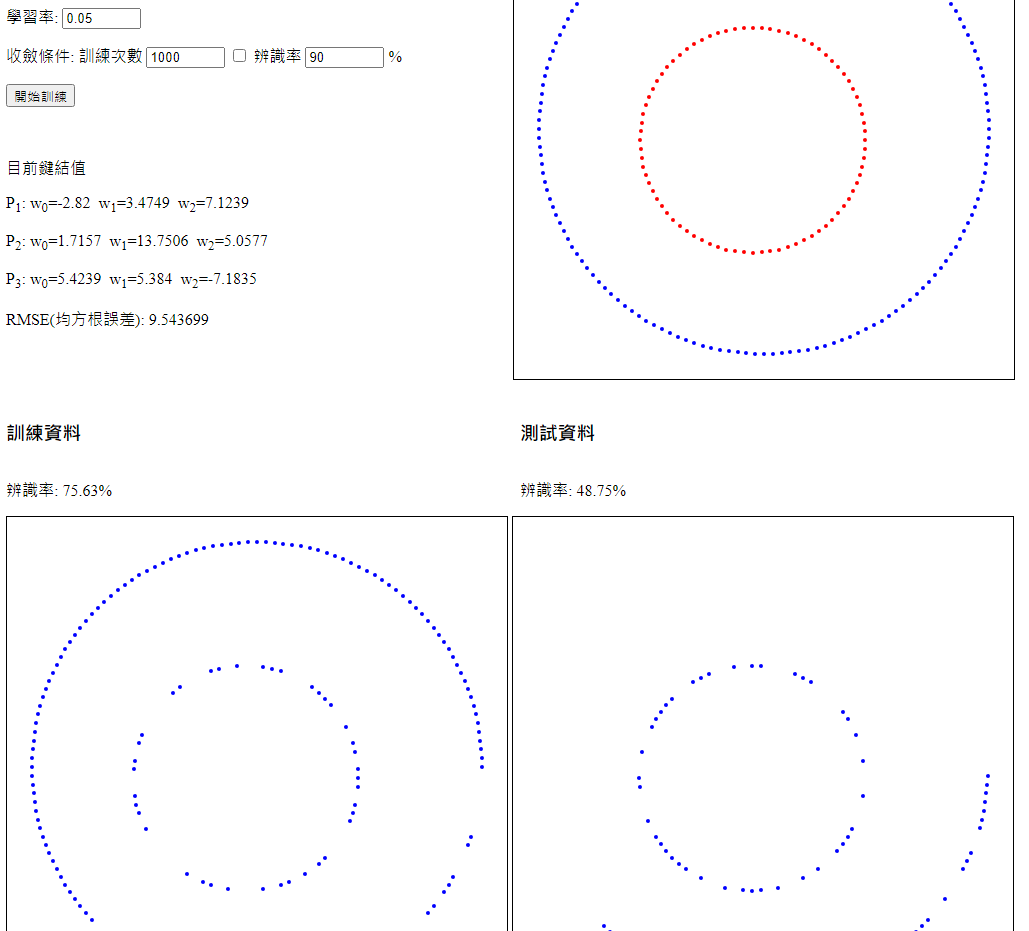
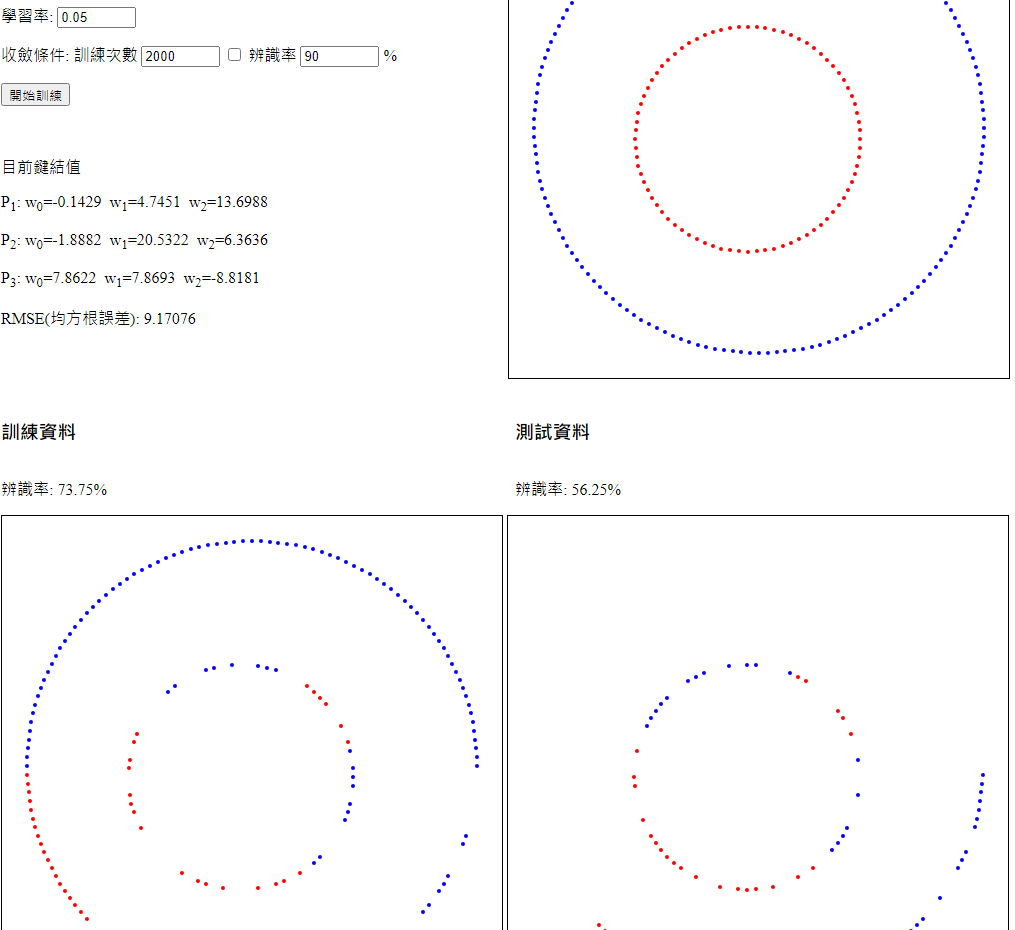
* **2Ccircle1.txt**

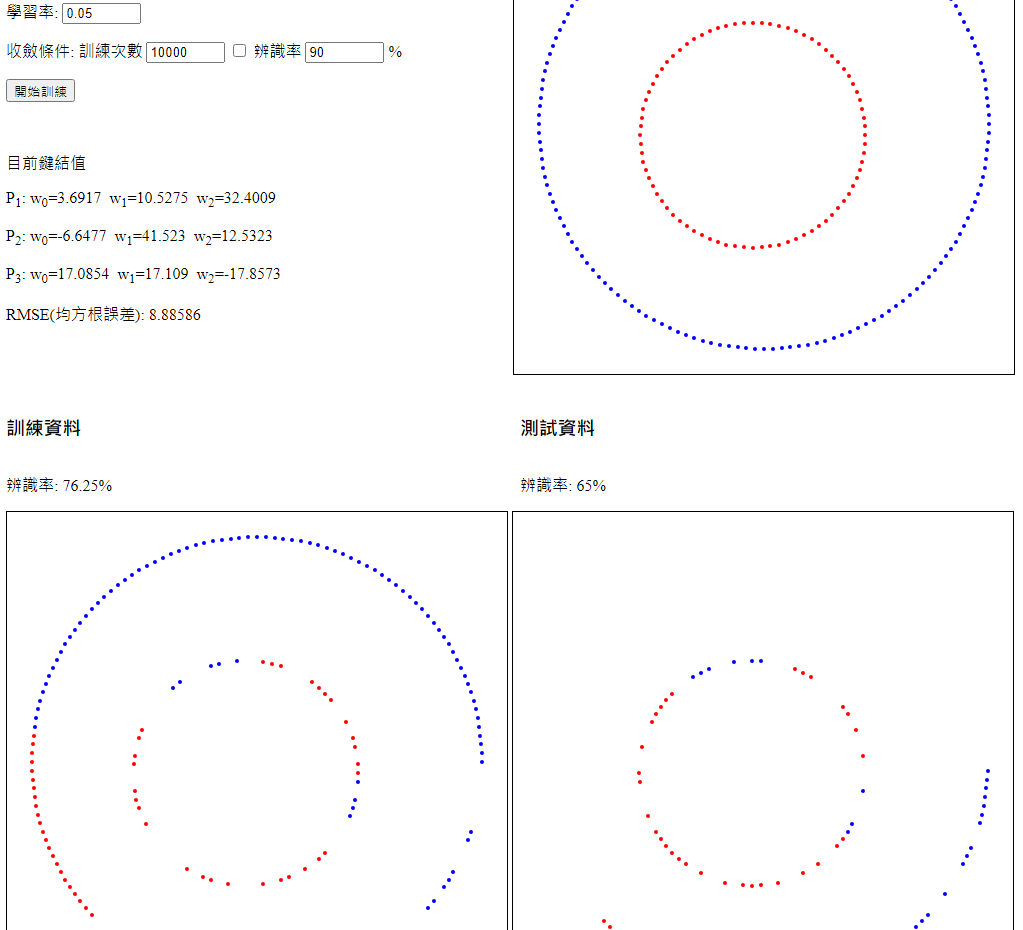
**初始畫面**



**訓練結果**



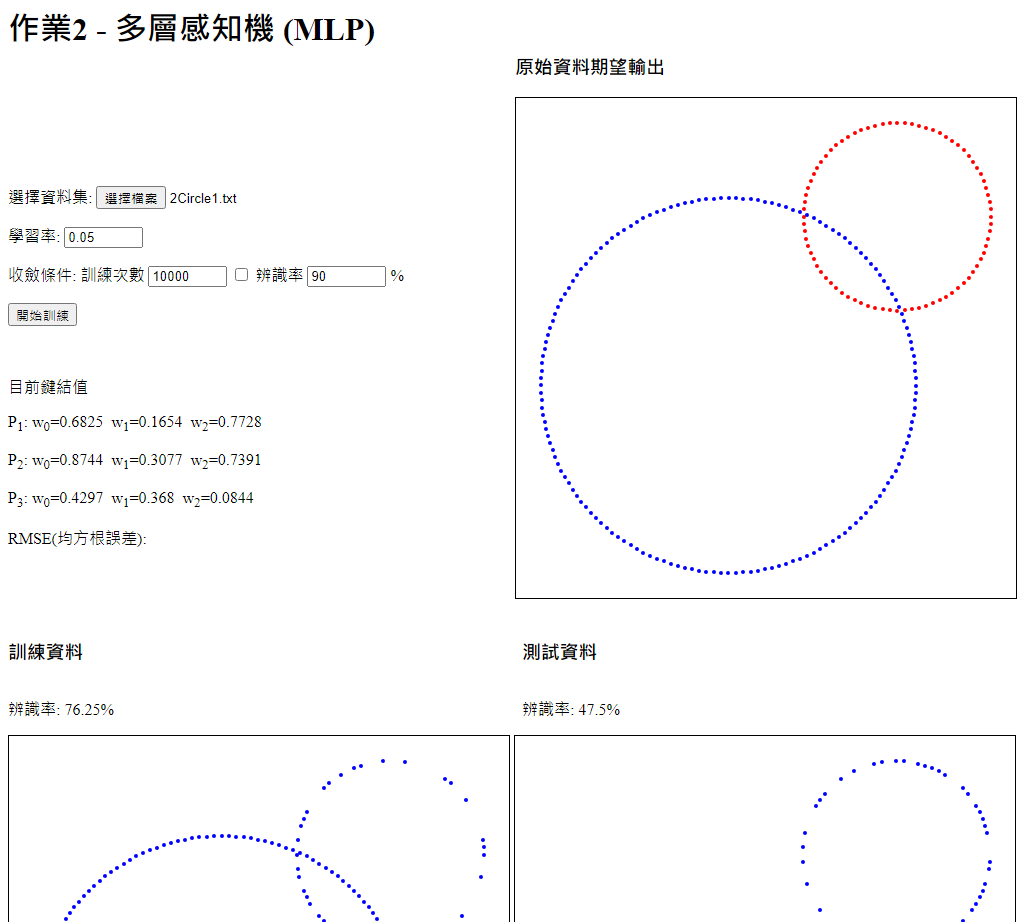


**說明**

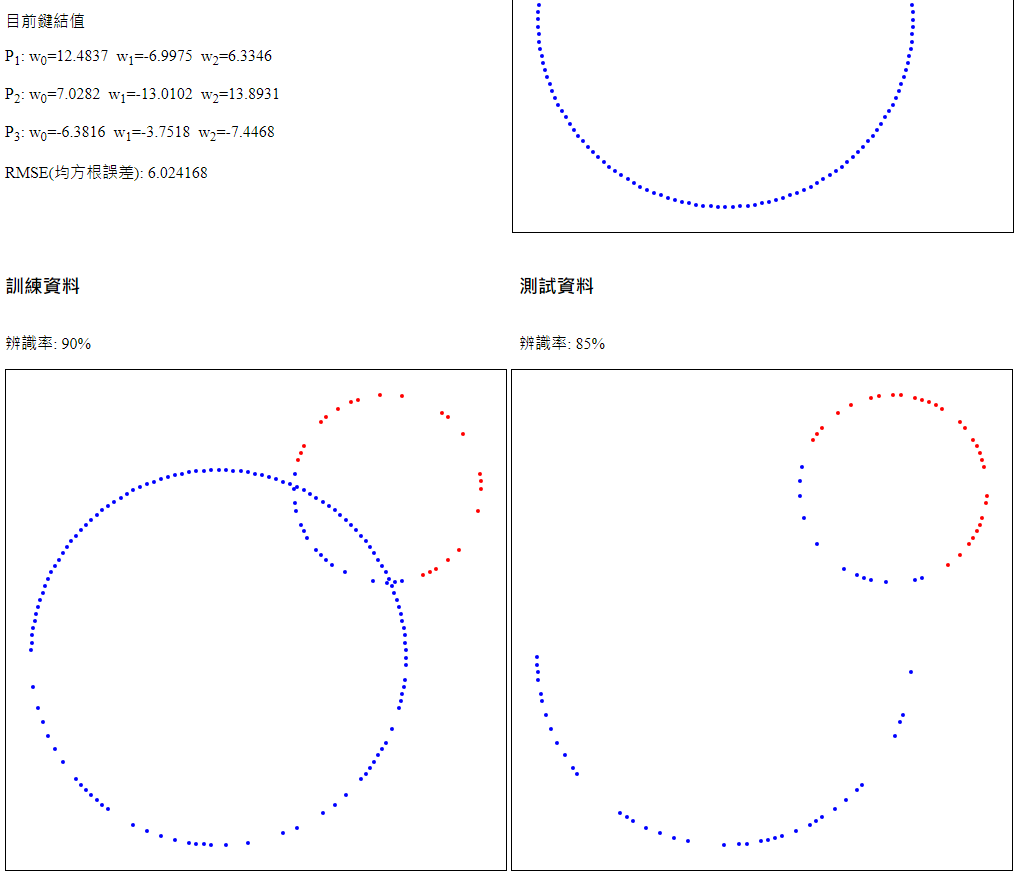
一開始一樣只調整學習率，設為0.5，辨識率從56.25%提升到75.63%。後來把學習率改成0.05，訓練次數改成1000，辨識率並沒有提升。接著再把訓練次數增加到2000，辨識率反而還下降到73.75%，我推測應該是學習率太小，導致他還沒學到比較好的鍵結值。最後訓練次數增加到10000時，辨識率還有提升到76.25%。

* **2Circle1.txt**

**初始畫面**



**訓練結果**

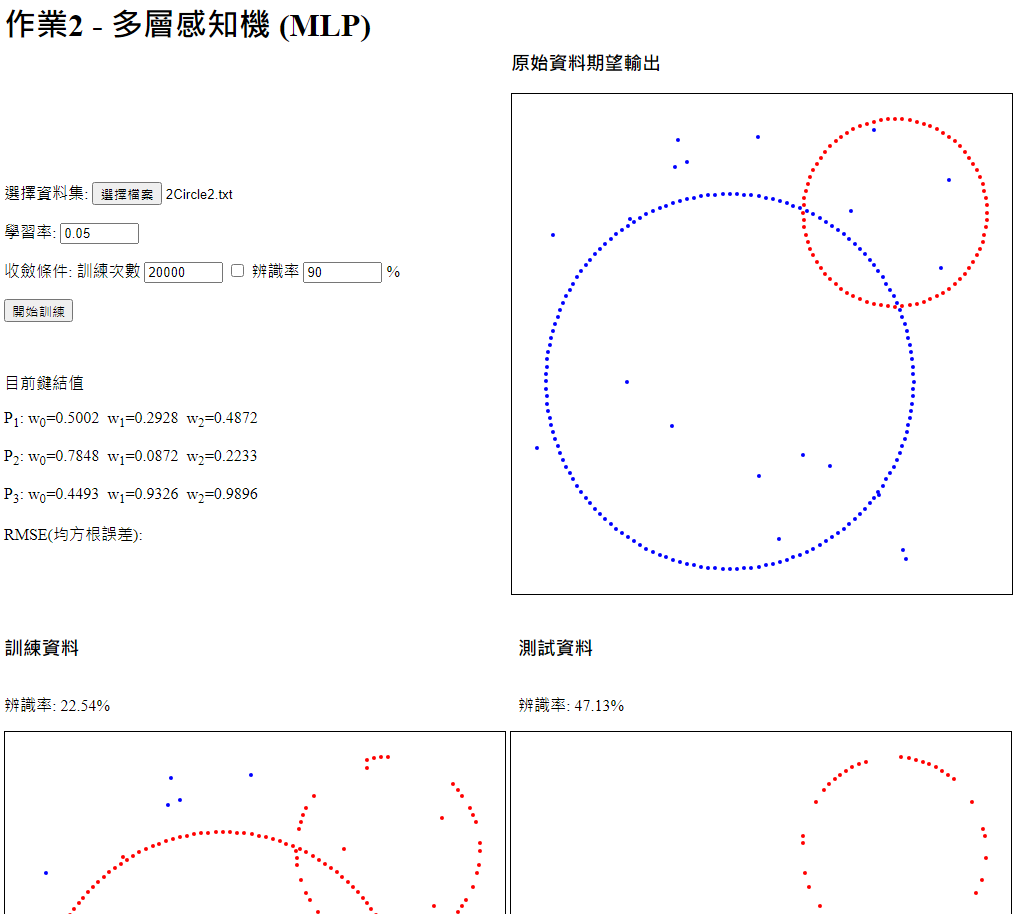


**說明**

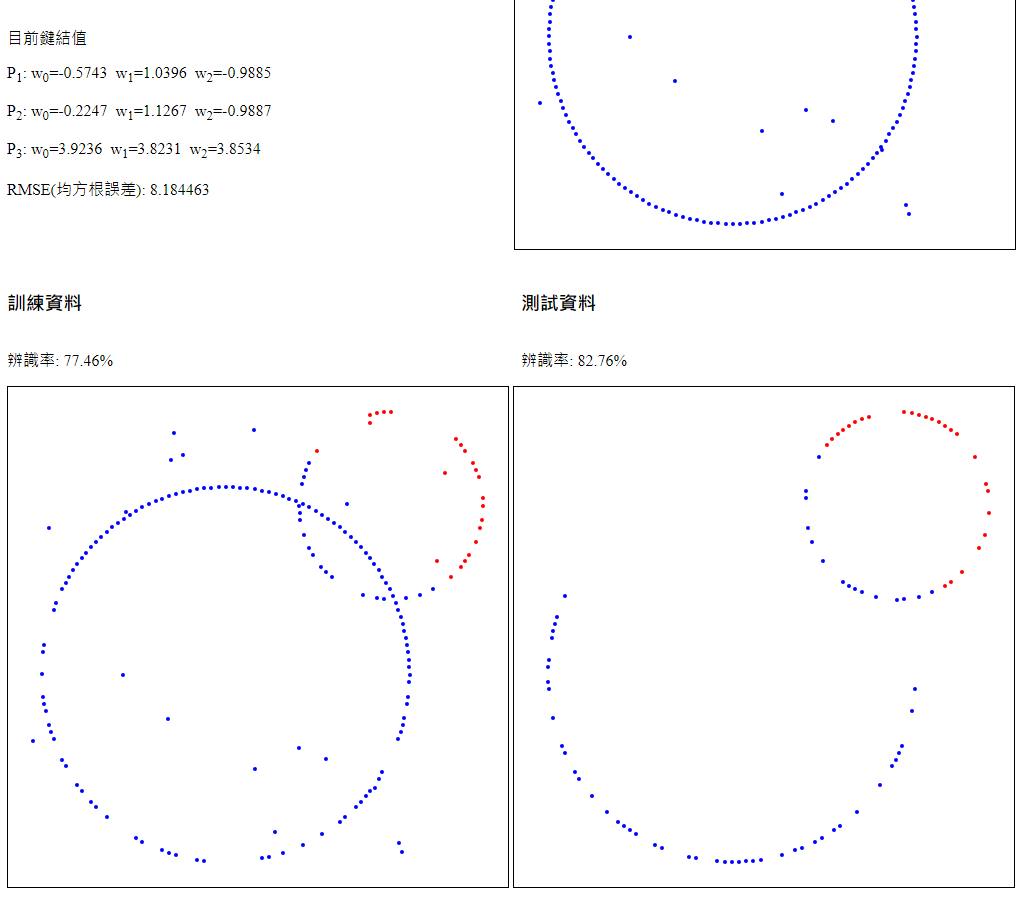
學習率設0.05，訓練次數設10000，辨識率可達到90%。後來幾次調整學習率以及訓練次數，雖然沒有提升辨識率，但RMSE有逐漸降低。

* **2Circle2.txt**

**初始畫面**



**訓練結果**

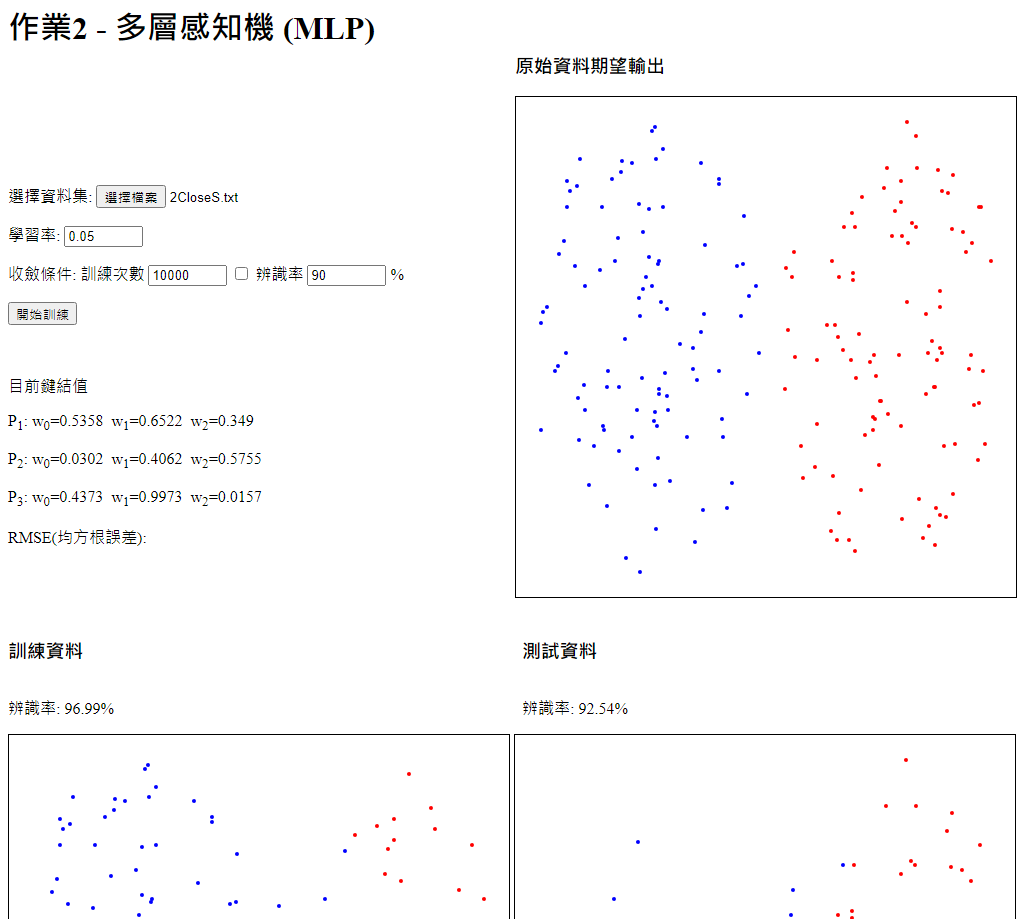


**說明**

學習率設0.05，訓練次數設10000，可能因為還有一些第三類資料，導致準確率又更低了一些，但還是可以到77.46%。

* **2CloseS.txt**

**初始畫面**



**訓練結果**

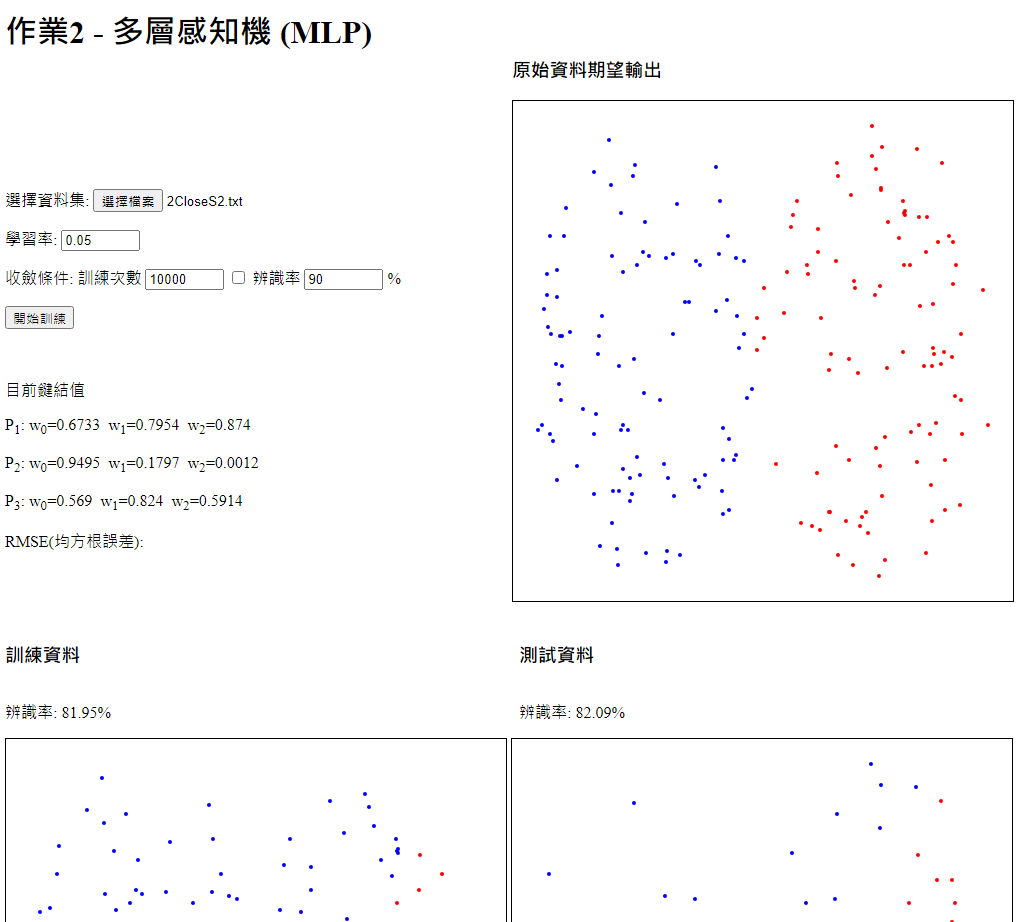


**說明**

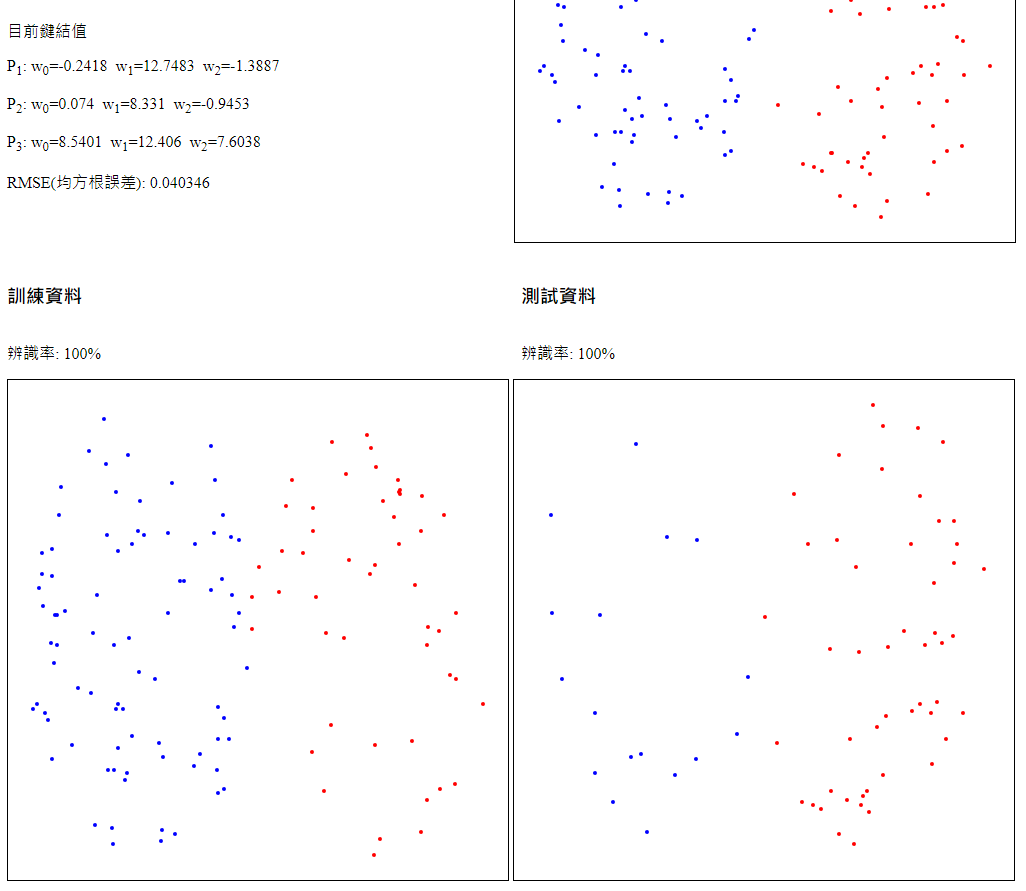
學習率設0.05，訓練次數設10000，一開始初始化很幸運地就有96.99%的辨識率，所以直接訓練就到100%。

* **2CloseS2.txt**

**初始畫面**

****

**訓練結果**

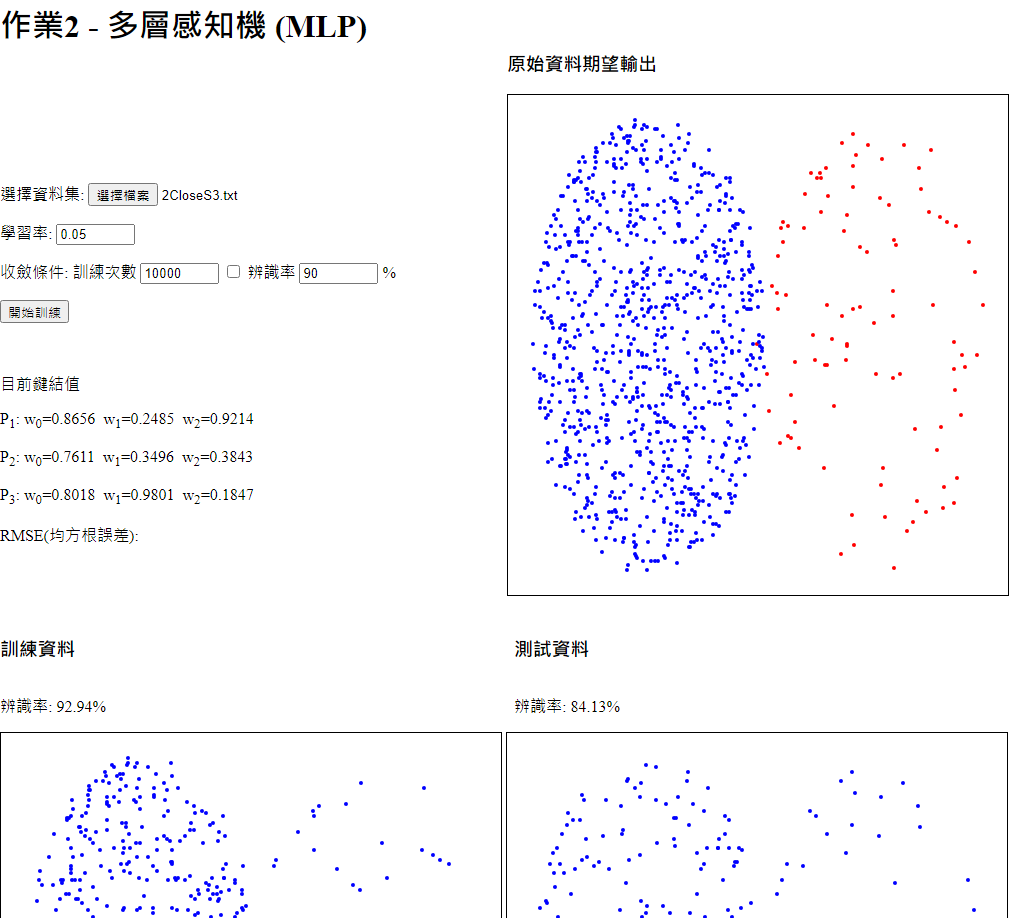


**說明**

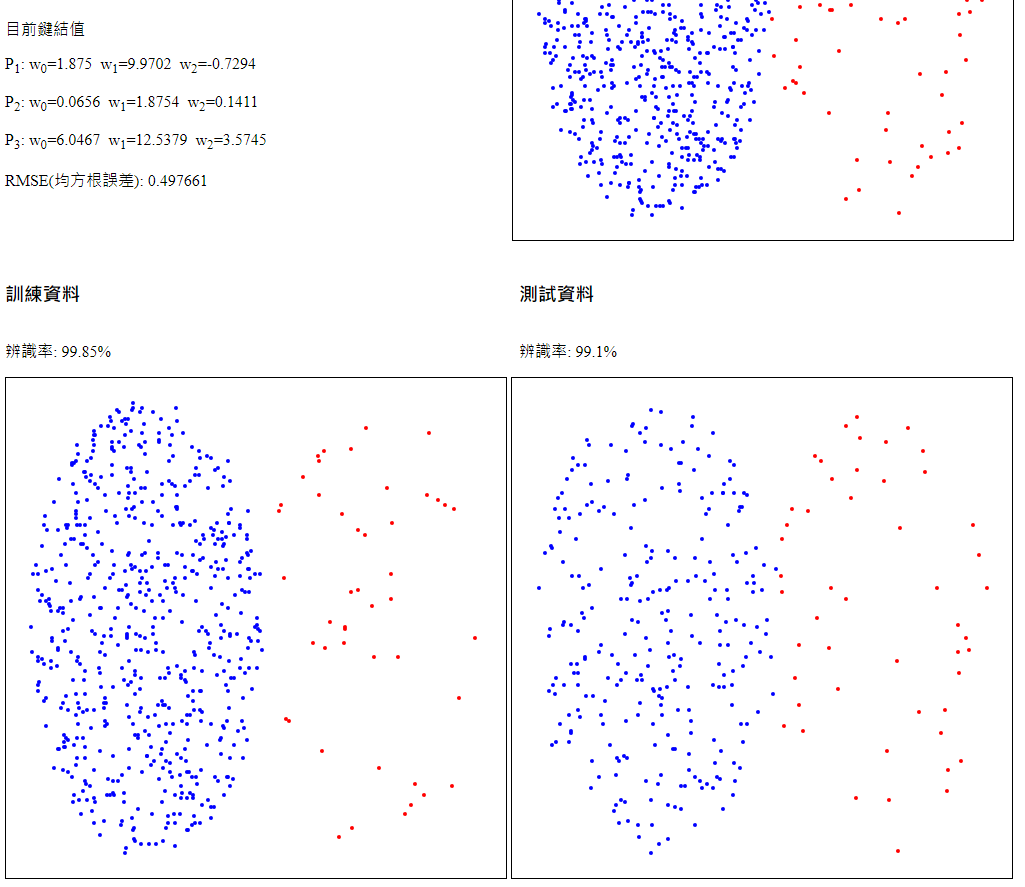
學習率設0.05，訓練次數設10000，結果兩邊都能成功分類，辨識率100%。

* **2CloseS3.txt**

**初始畫面**



**訓練結果**

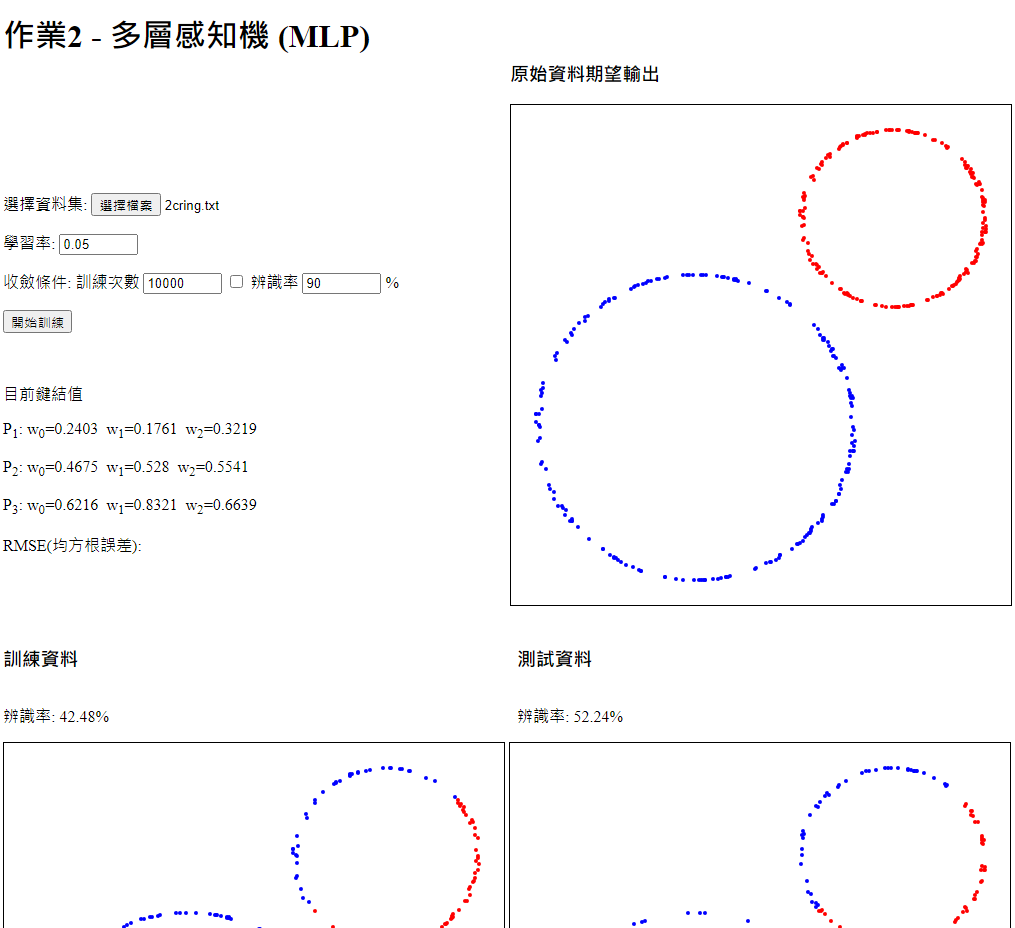


**說明**

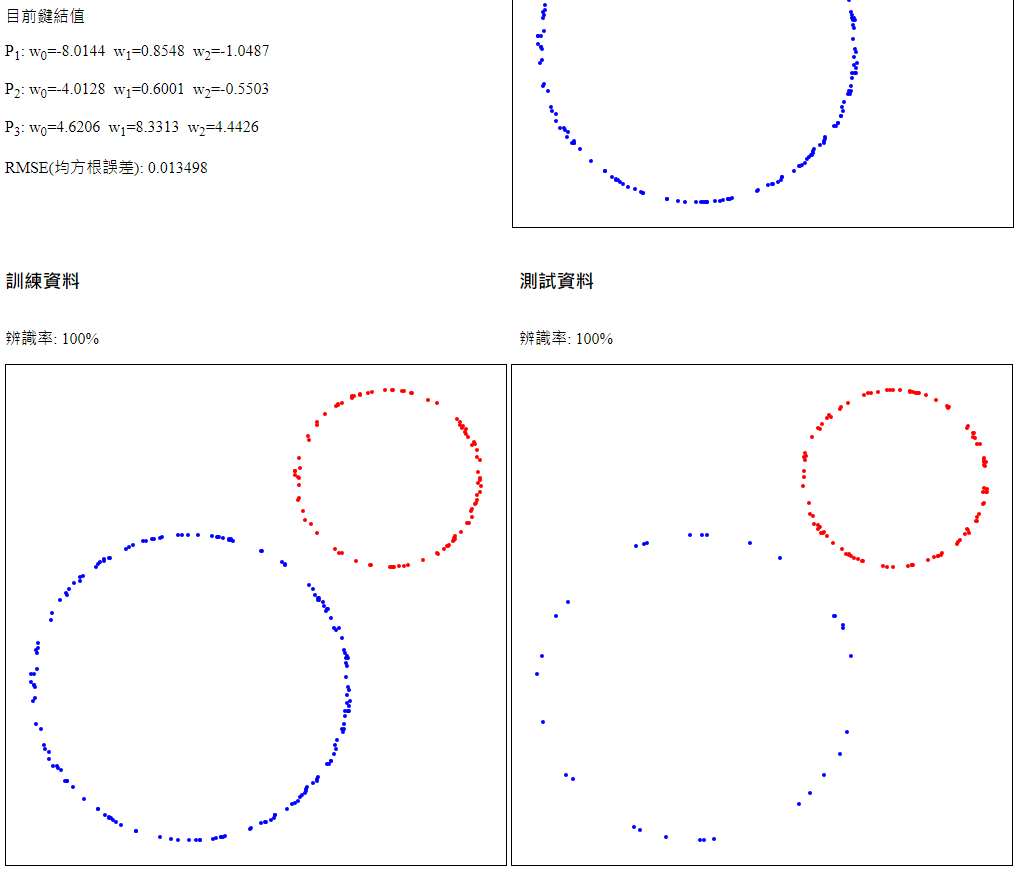
學習率設0.05，訓練次數設10000，訓練和測試資料都有99%辨識率。

* **2cring.txt**

**初始畫面**



**訓練結果**

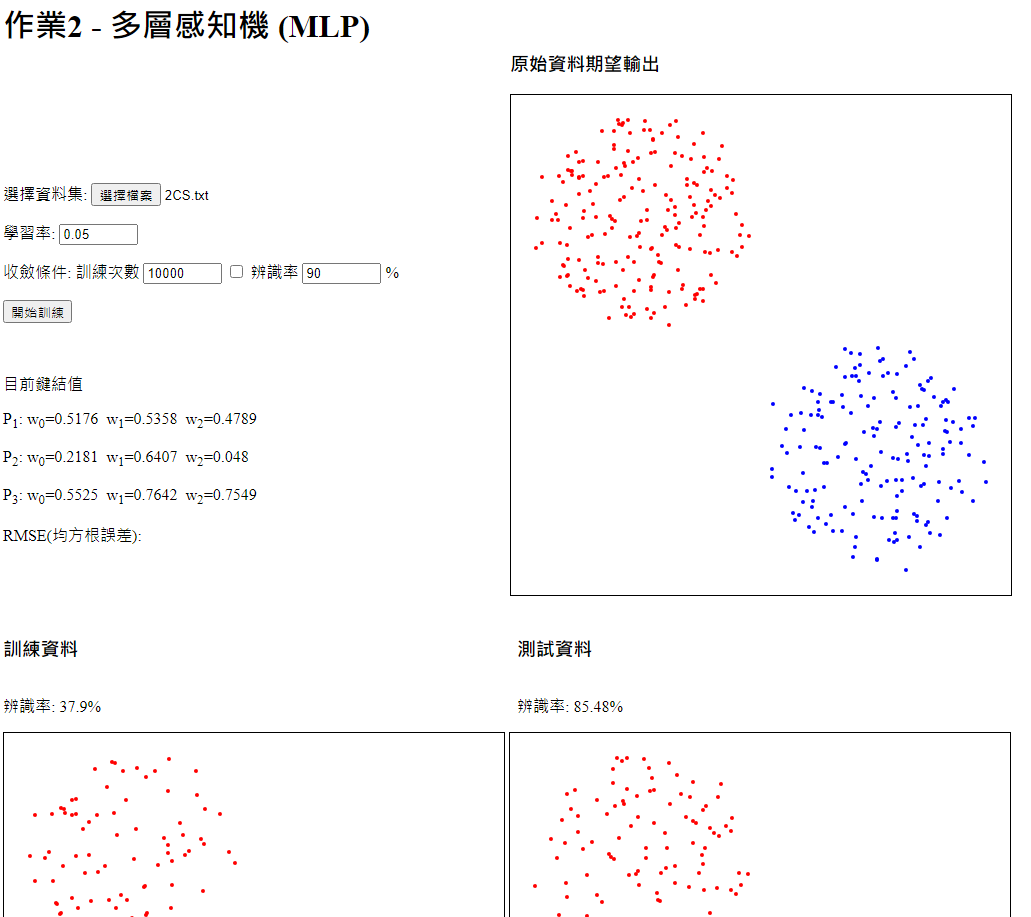


**說明**

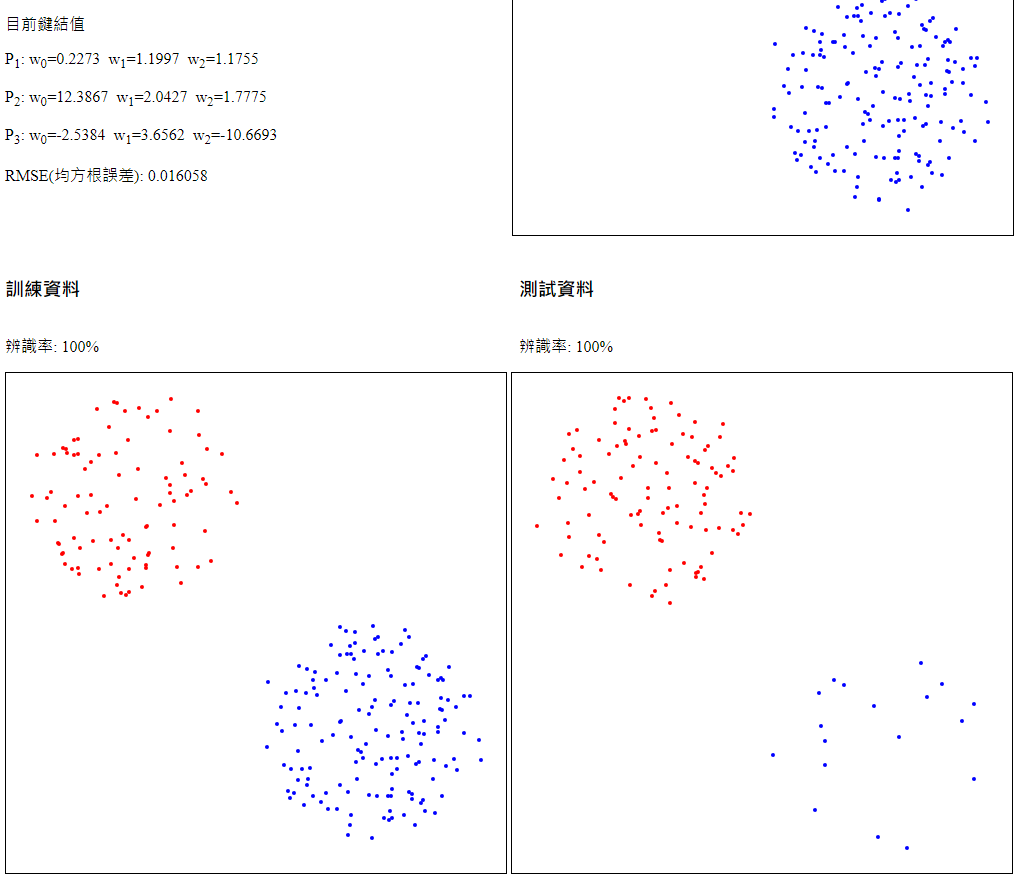
學習率設0.05，訓練次數設10000，兩邊都達到100%。

* **2CS.txt**

**初始畫面**



**訓練結果**

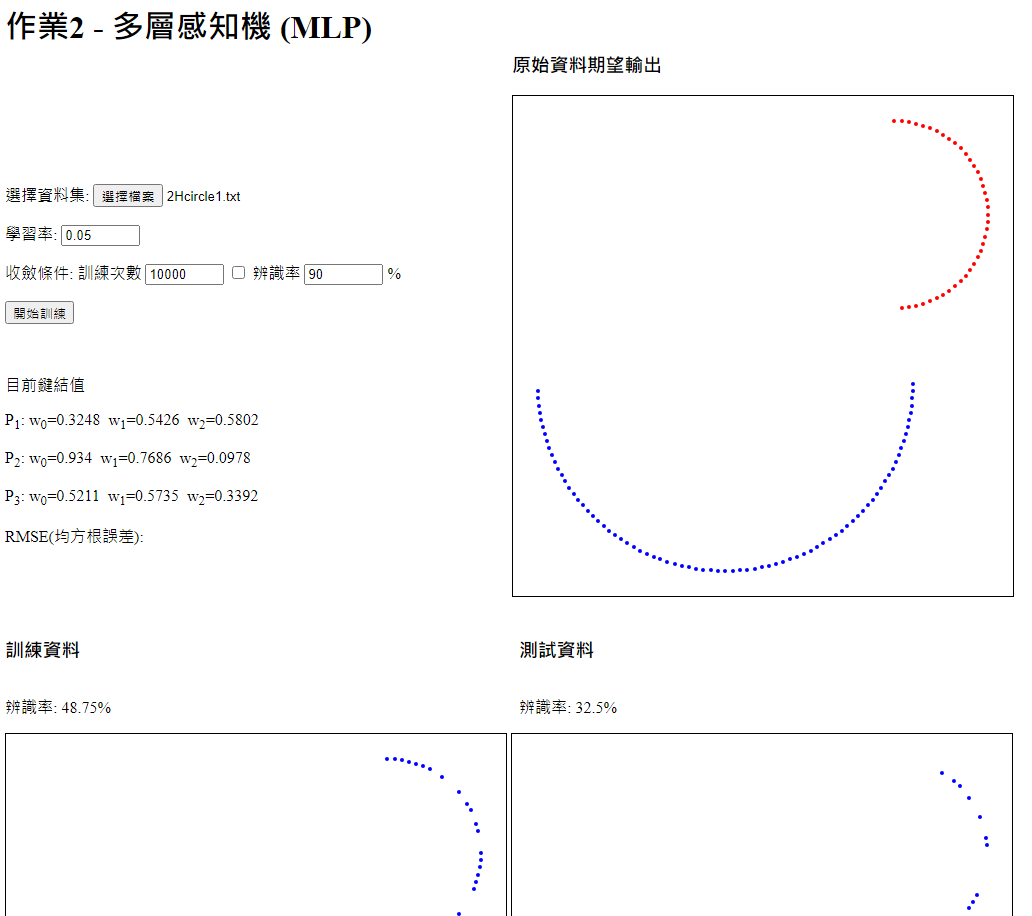


**說明**

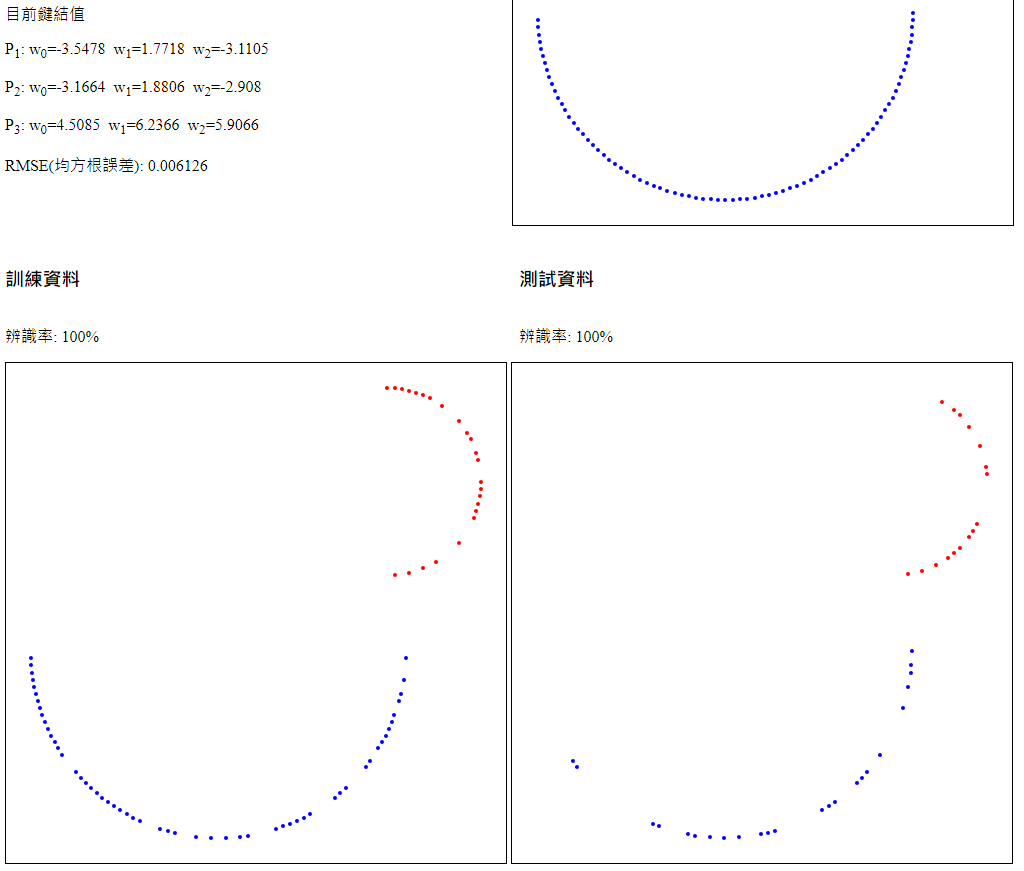
學習率設0.05，訓練次數設10000，兩邊都達到100%。

* **2Hcircle1.txt**

**初始畫面**



**訓練結果**

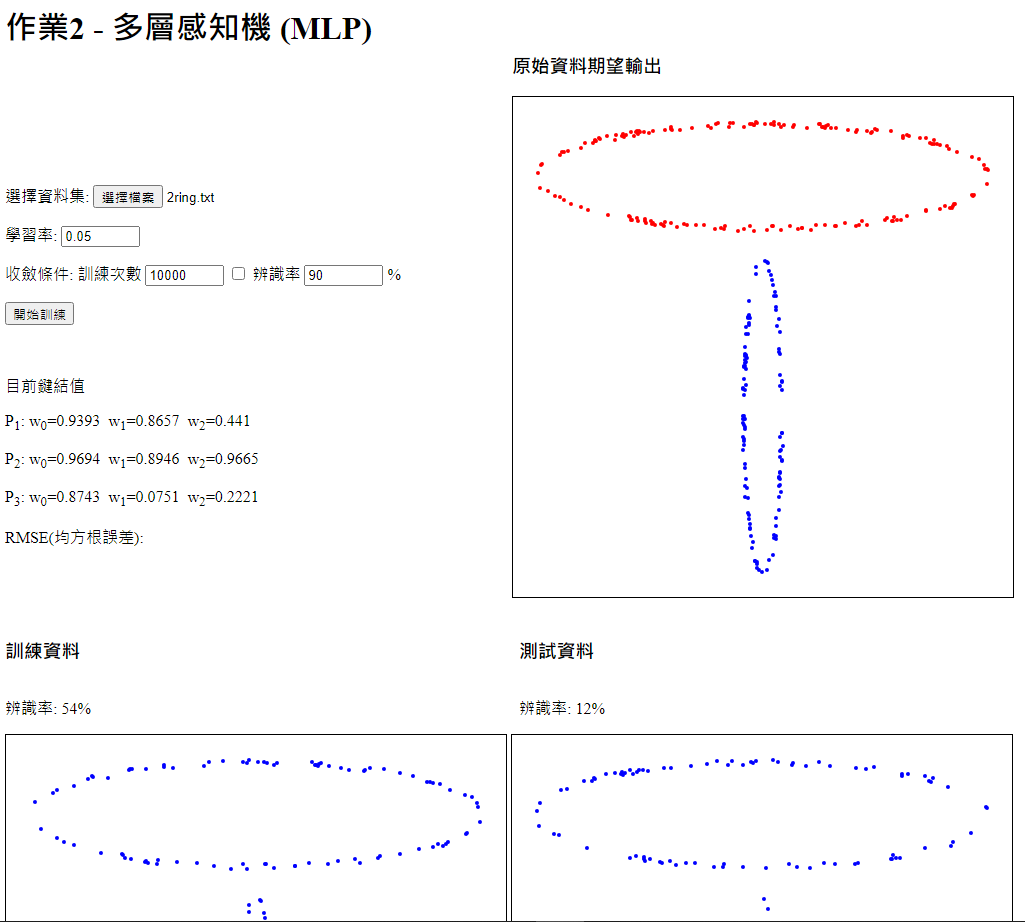


**說明**

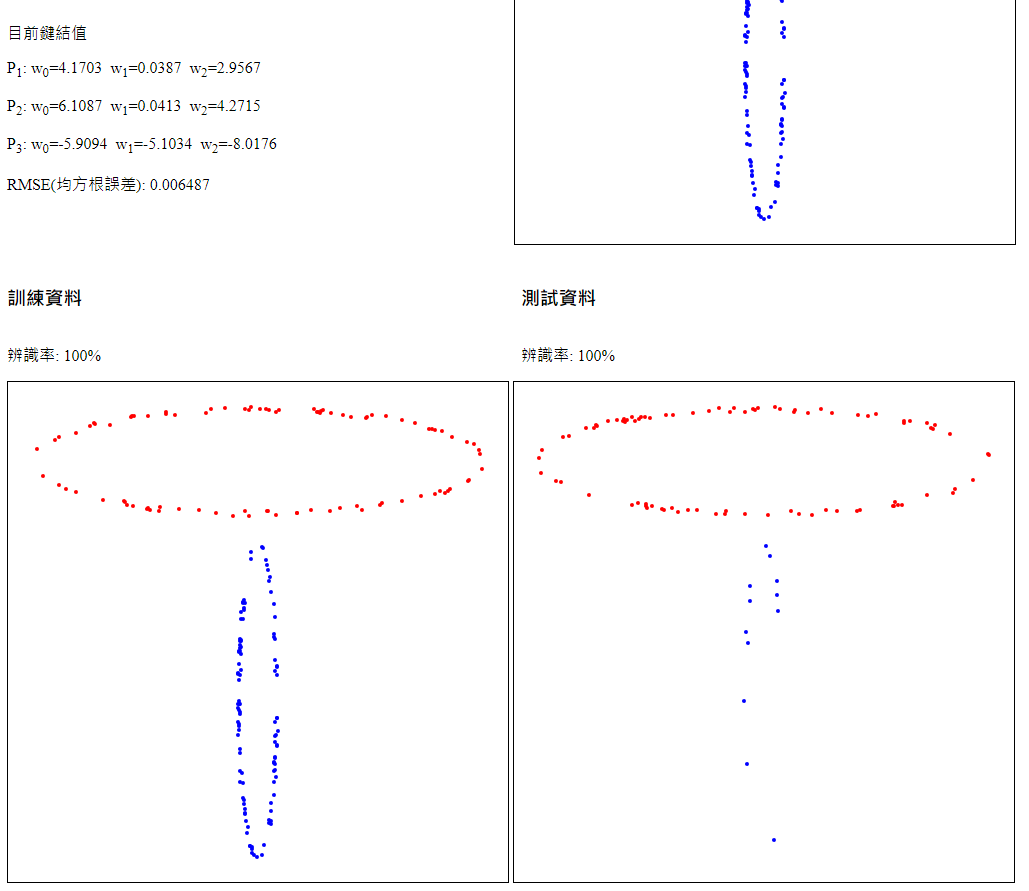
學習率設0.05，訓練次數設10000，兩邊都達到100%。

* **2ring.txt**

**初始畫面**



**訓練結果**



**說明**

學習率設0.05，訓練次數設10000，兩邊都達到100%。

1. **實驗結果分析及討論**

由上面的實驗可以發現，可線性分割的資料集對於多層感知機來說都很容易分類，幾乎都可以達到100%的辨識率，訓練的困難度取決於這兩類的資料靠得有多近，靠得愈近訓練起來愈困難，從2CloseS、2CloseS2、2CloseS3可以觀察到這個現象。而有些訓練辨識率已經100%，但測試辨識率還沒到100%的情況，例如perceptron2，這是因為測試資料沒有學習到的關係，之後可以針對這點去改善，在每次學習循環結束時，重新隨機抽取新的訓練資料，讓學習變得更全面。另外比較特殊的情況2Ccircle1，調整了很多次參數，辨識率就卡在76%無法提升了，我推測應該是我的網路架構太簡單，無法應付這類型的資料集，也許下次在隱藏層多放幾個神經元就能解決。