類神經網路作業二 109522055 張至妤 資工碩一

1. 程式簡介-多層感知機

此程式實作多層感知機架構，dataset及變數輸入完後即可進行運算，運算前會先將dataset隨機2/3當訓練資料、1/3當測試資料，並隨機產生初始鍵結值。

多層感知機架構部分：

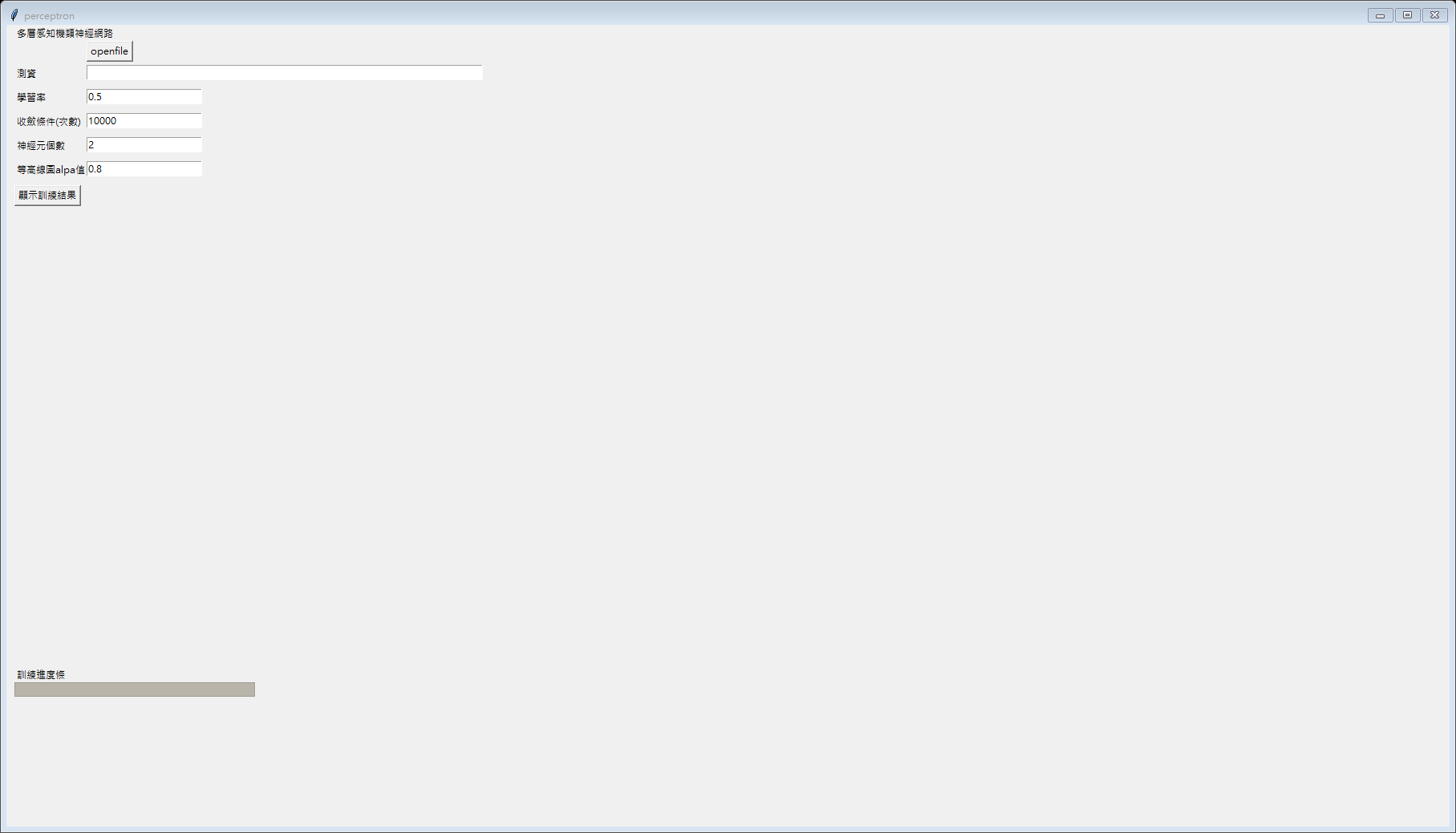
1. 會將收斂次數除以training data筆數，以餘數當訓練第幾筆資料，也就是說當收斂次數為100而training data共10筆則每筆訓練10次，因此我的收斂次數設定看似較大，此方法是為了跟第一次作業做比較(因單層感知機的投影片範例為此設置)，例：第n次即訓練第n筆資料非所有資料都跑n次。
2. 前饋階段：在第n筆訓練資料時將輸入向量送入網路，因我實作多神經元，所以根據輸入神經元的個數依序算出隱藏層中每個類神經元的輸出，最後算出輸出層類神經元的輸出。
3. 倒傳遞階段：先算位於輸出層的類神經元在第n筆學習時的區域梯度函數，再以倒傳遞的方式一層一層計算出每個隱藏層的類神經元的區域梯度函數。
4. 調整鍵結值：根據區域梯度函數修正鍵結值，並重複直到使用者輸入的收斂次數。

最後再根據多層感知機算出的回傳值畫圖及顯示文字資料。因訓練時間較久所以我有顯示進度條表示訓練的部分完成比例。再者，等高線圖產生時間沒這麼快，所以當圖都完全畫好且準確率及鍵結值都計算顯示完，視窗左下角會顯示DONE表示程式跑完，讓使用者知道程式是否跑完。

以下為程式各function簡介：

* def do\_openfile():
  + GUI上讀取測試資料的按鈕，當按下後即呼叫此程式回傳測資路徑檔名。
* def readtext(filename):
  + 讀檔，回傳成雙列表的格式。
* def preprocess(data):
  + 處理資料，隨機分2/3為training data；1/3為test data。
* def sig(value):
  + 在perceptron function中運算活化函數(sigmoid)時呼叫的。
* def perceptron(train, test, learning\_rate, maxtime, initial\_weight, theta, neuronum):
  + 進行訓練及測試，計算準確率、RMSE及鍵結值。並特徵空間的座標轉換。
  + 最後再分別呼叫畫圖程式，並回傳前面計算的文字顯示結果。
* def make\_meshgrid(x, y, h=.02):
  + 在plotData2D function中呼叫計算網格點，使得可畫出等高線圖。
* def plotData2D(train, test, w, num):
  + 傳入training data, testing data, 鍵結值以及畫第幾張圖的參數。
  + 當num=1畫原始資料集，num=2畫特徵空間圖，num=3畫等高線圖
* def enter():
  + GUI上輸入完測資、學習率及收斂條件後按”顯示訓練結果”按鈕即呼叫此程式，再由此程式呼叫各副程式進行運算即顯示。
* main: GUI一開始即顯示的畫面設定，及控制按鈕呼叫副程式。

1. 程式執行說明



**進度條，可知訓練的進度(以收斂次數為底)**

**Toolbar，可放大圖或儲存，每張圖都各有一個**

**Button，按了後即開始運算並畫圖**

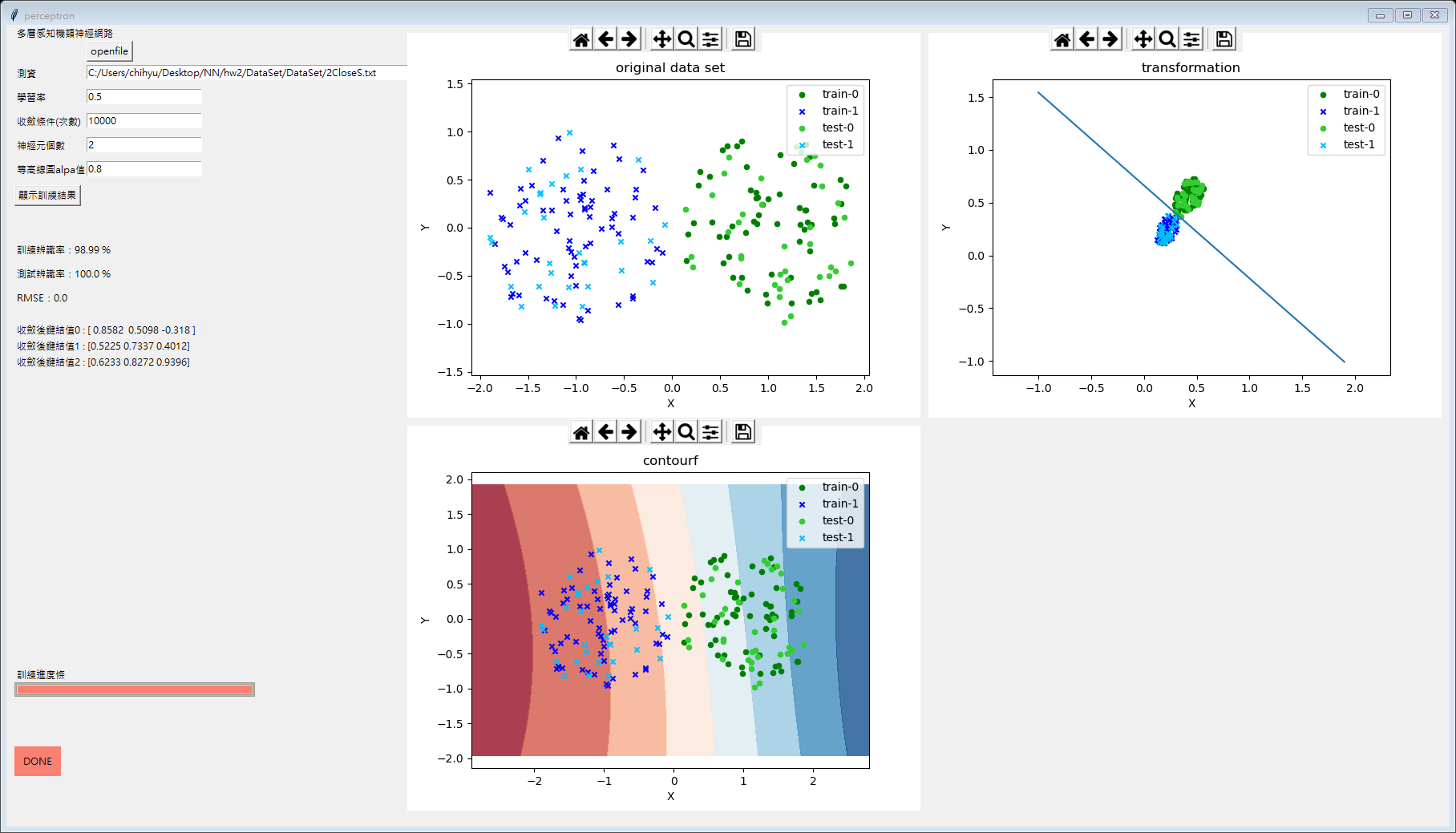
**設定學習率，預設為0.5**

**設定收斂次數，預設為10000**

**設定神經元個數，預設為2**

**設定等高線圖alpa值，預設為0.8**

**Button，按了後即跳出選取檔案畫面**



**進度條為訓練迴圈的進度，迴圈跑完即訓練完成**

**當全部圖都畫完也顯示完文字結果便會出現”DONE”**

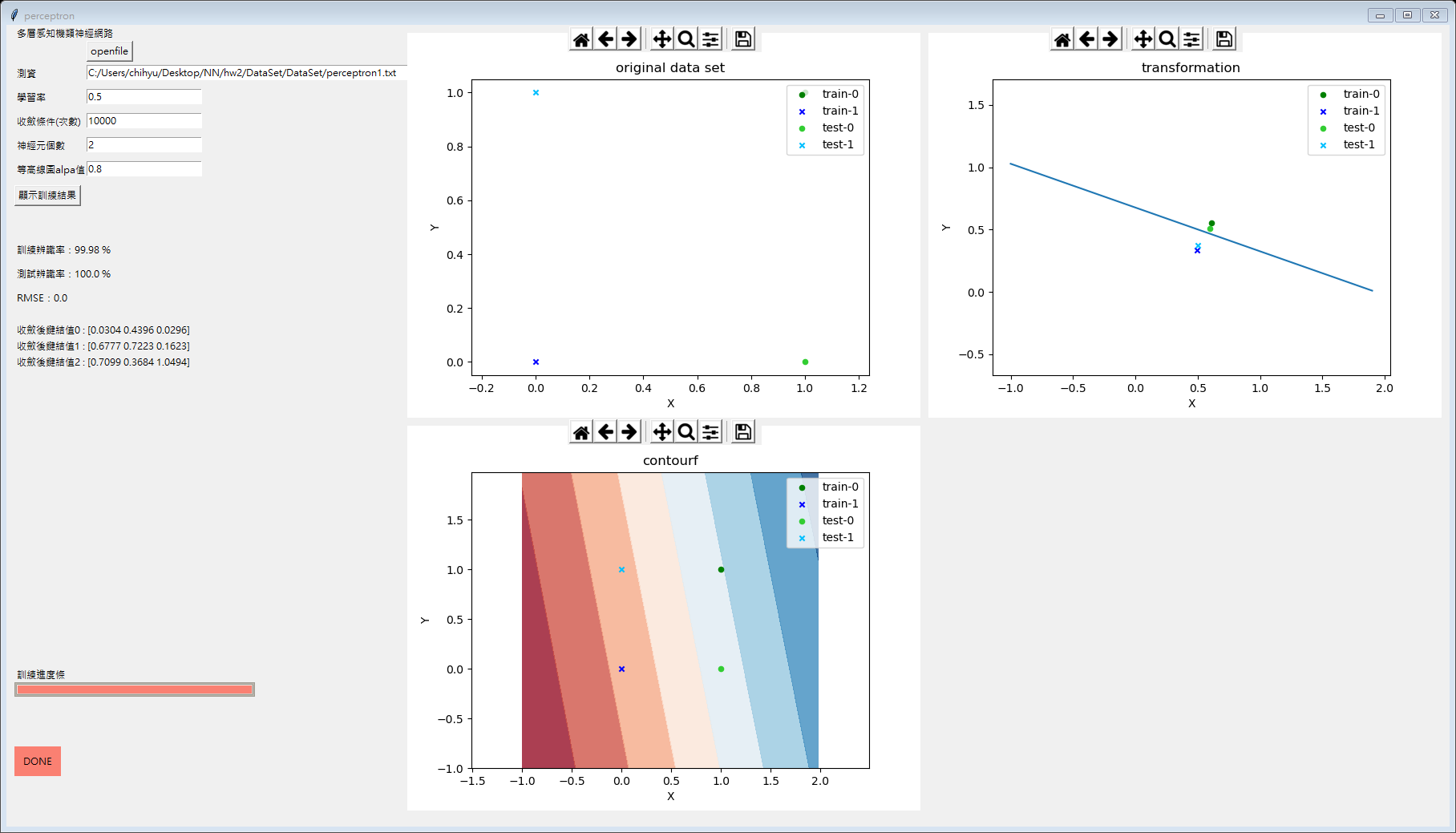
**運算完後顯示訓練辨識率、測試辨識率、RMSE及鍵結值**

**同label的圖示一樣並顏色相近**

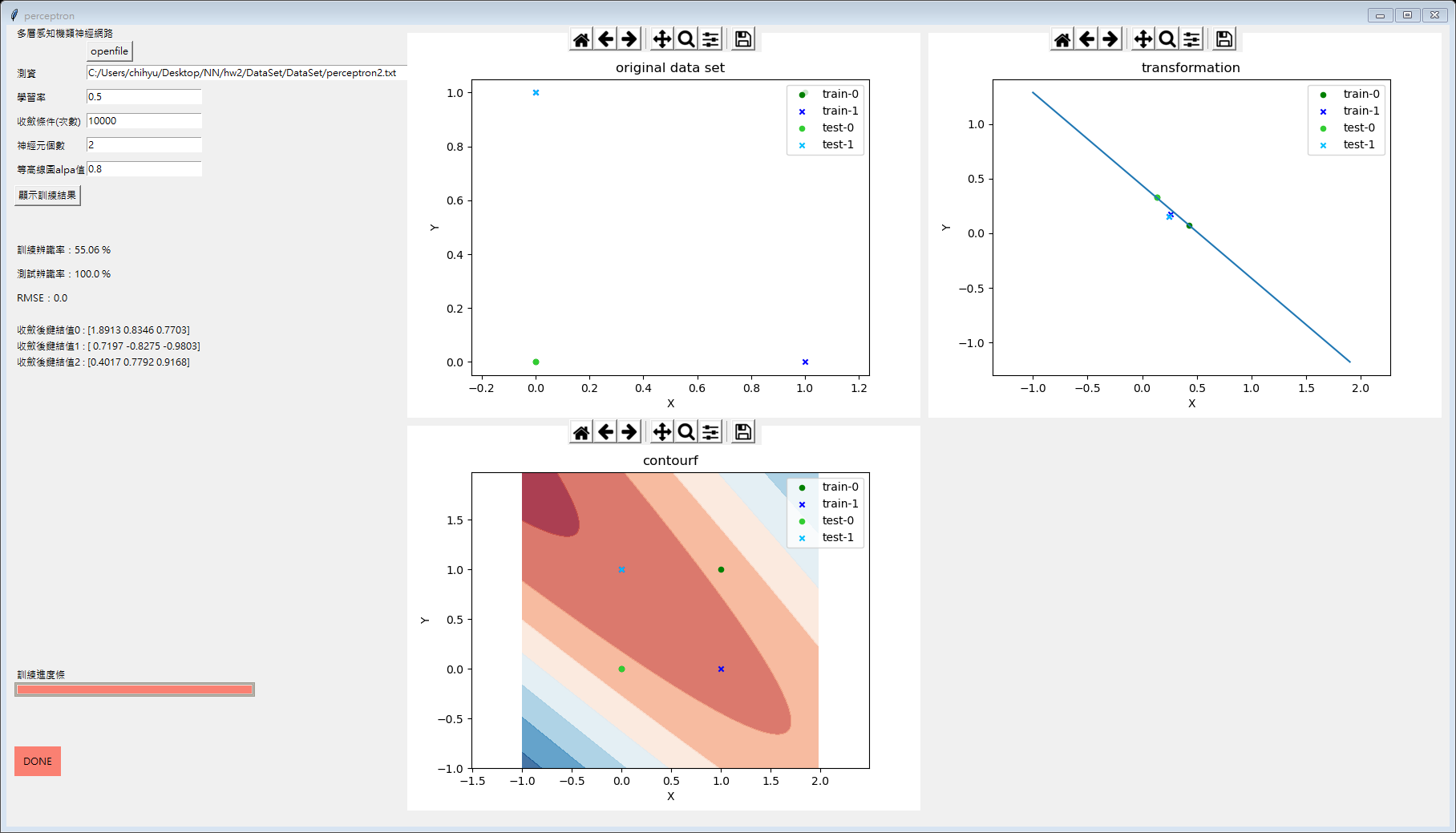
**Training data跟testing data分顏色顯示**

1. 實驗結果

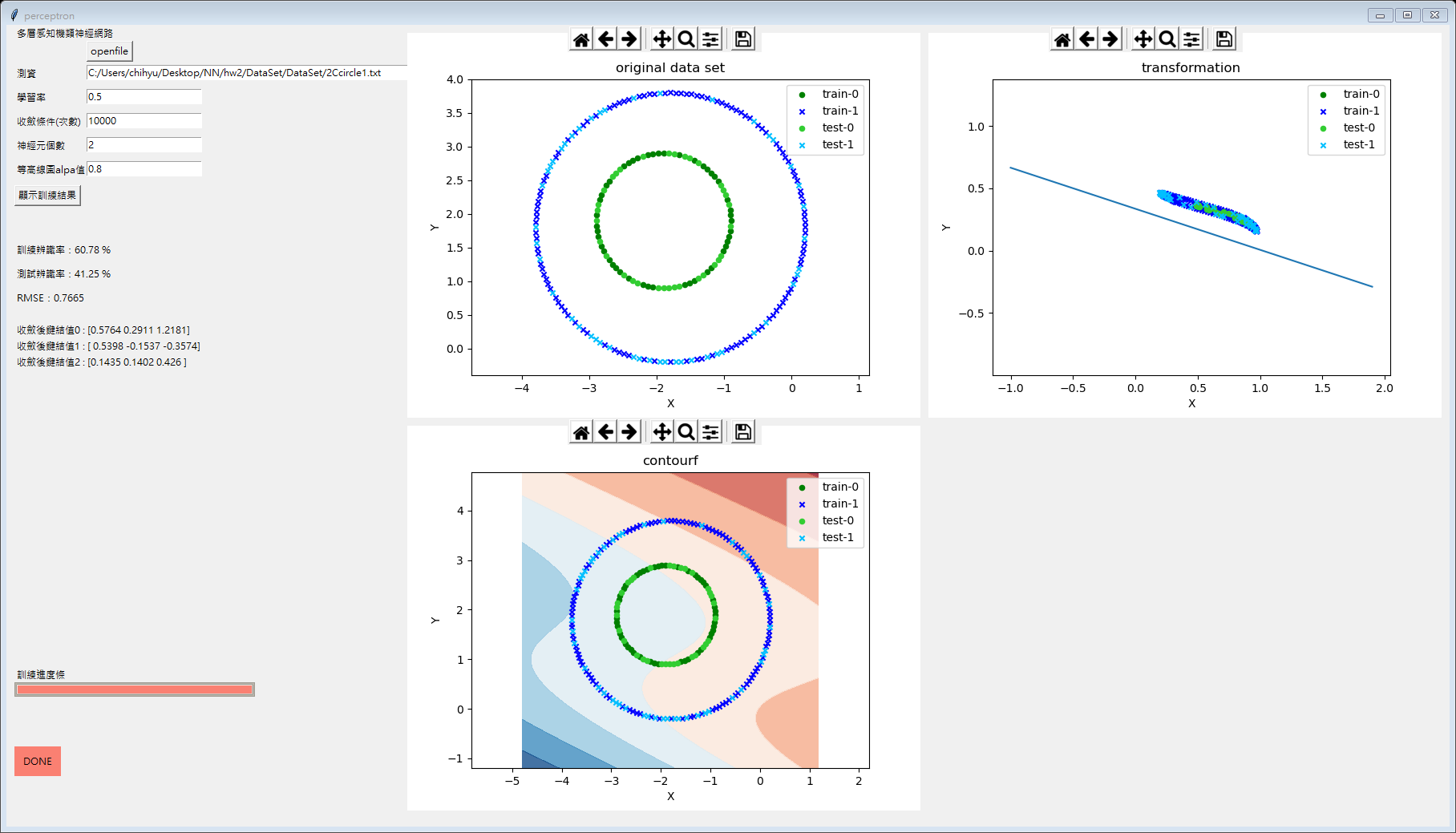
* perceptron1：可以準確的分類，透過等高線圖也看出無模糊的地帶。



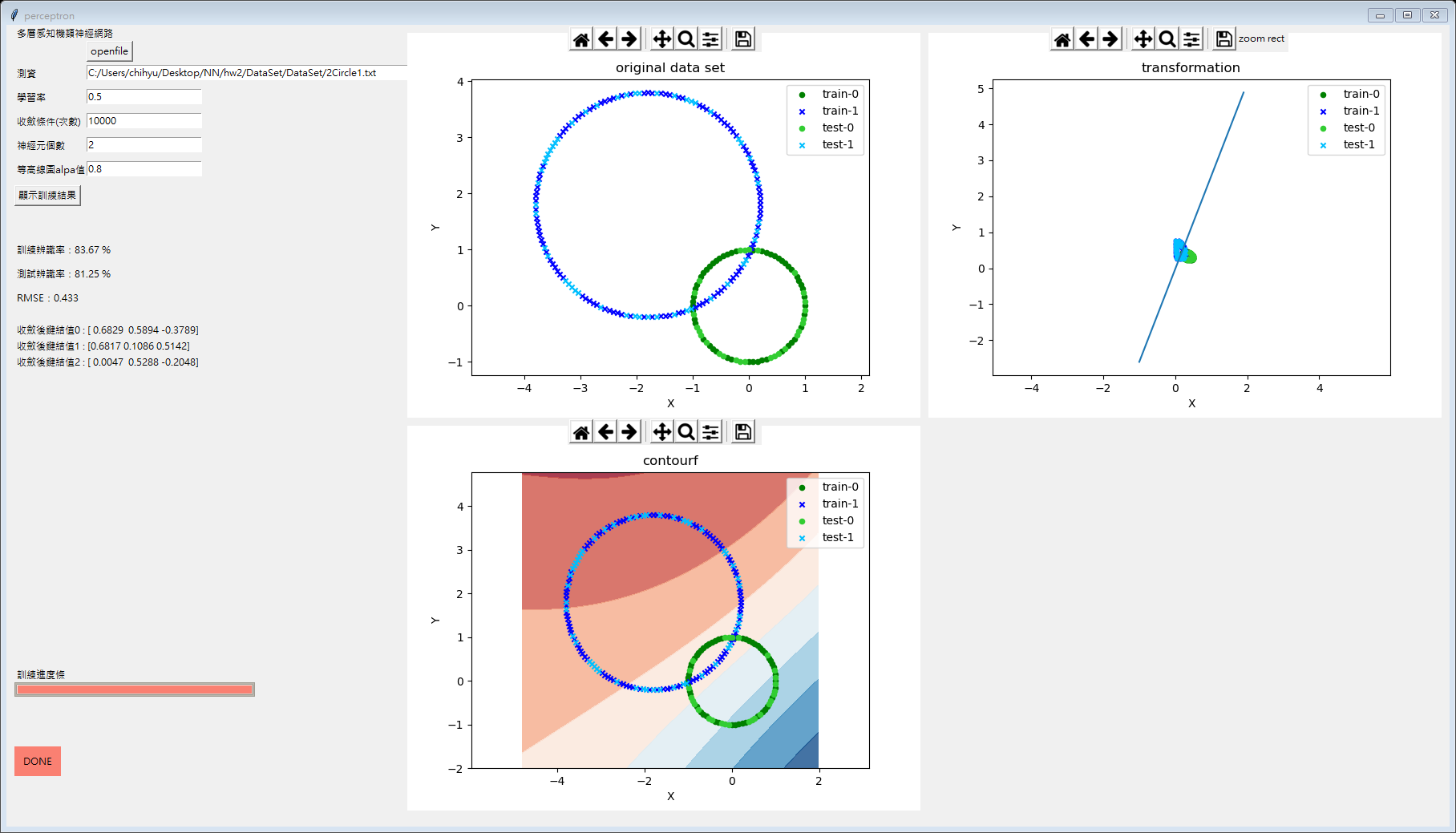
* perceptron2：因為這個測資為只有四筆如果只拿2/3去train，會使得資料無法被完善訓練，因此為了實作xor我把此資料的四筆皆拿去train即可得到此結果。



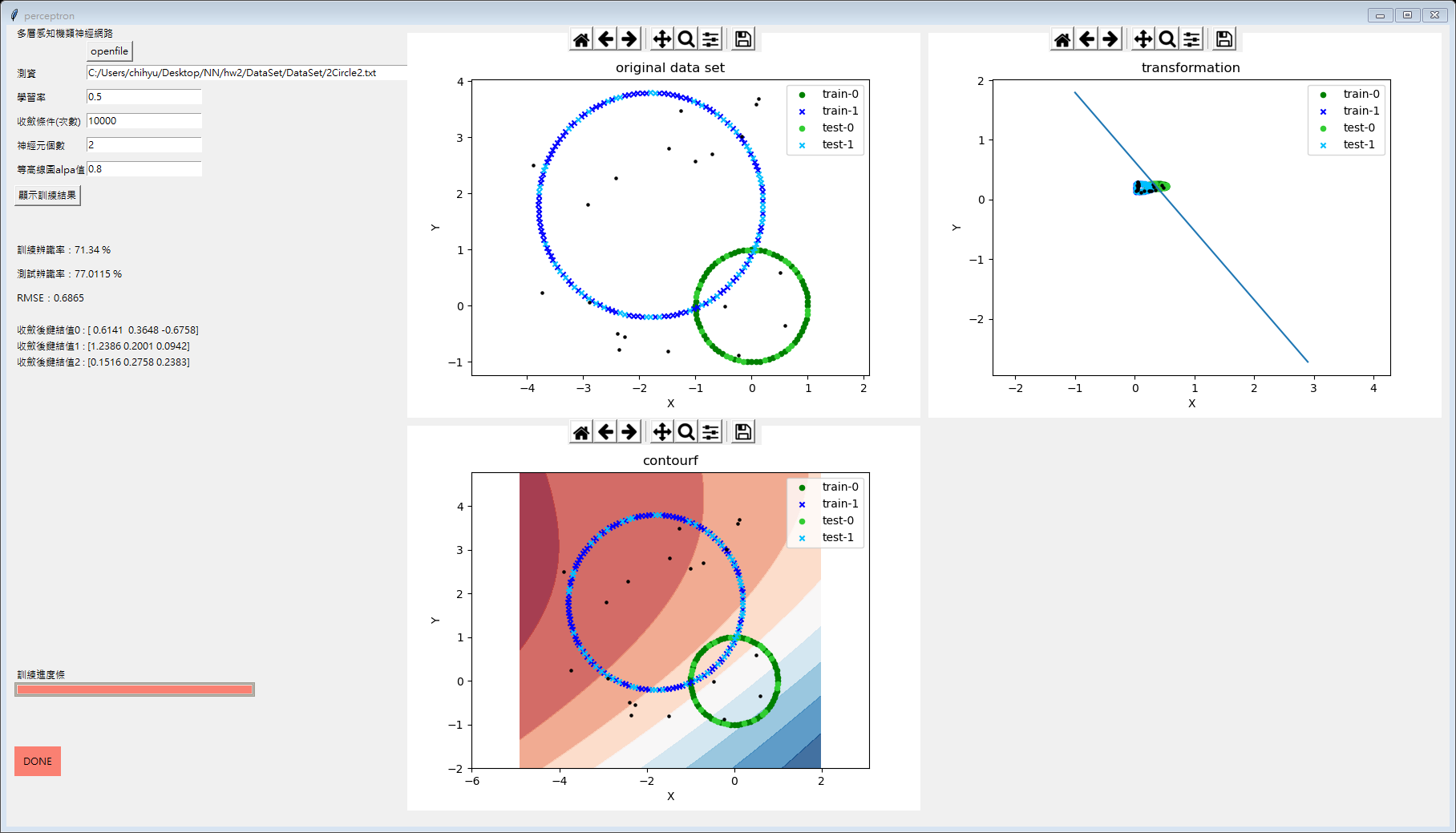
* 2Ccircle1：這個測資測試多次都較無法完善分割，或許要實作多隱藏層才能分割。



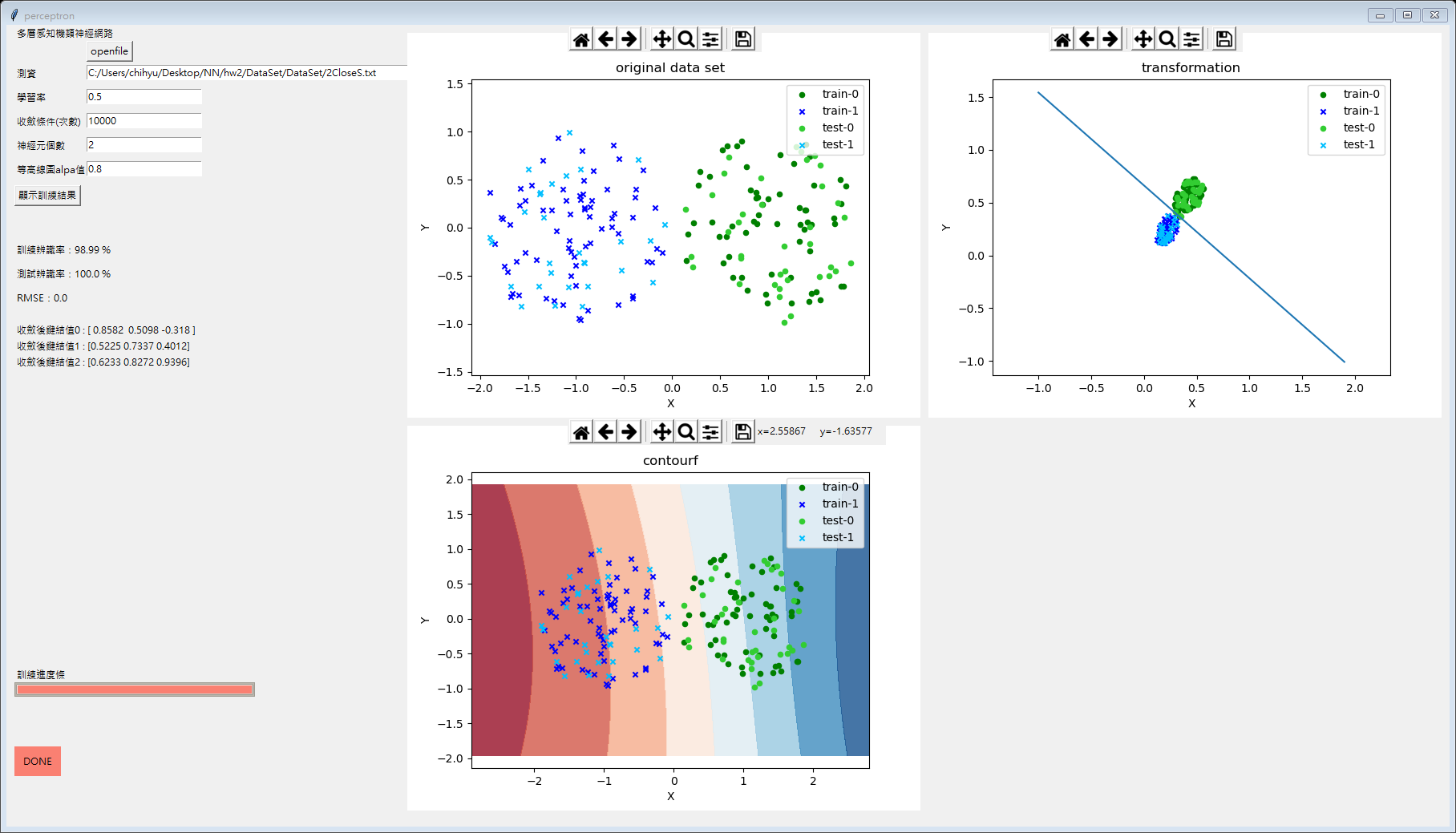
* 2Circle1：這個測資分割效果還不錯，能大致正確，只有中間區塊容易辨識錯。



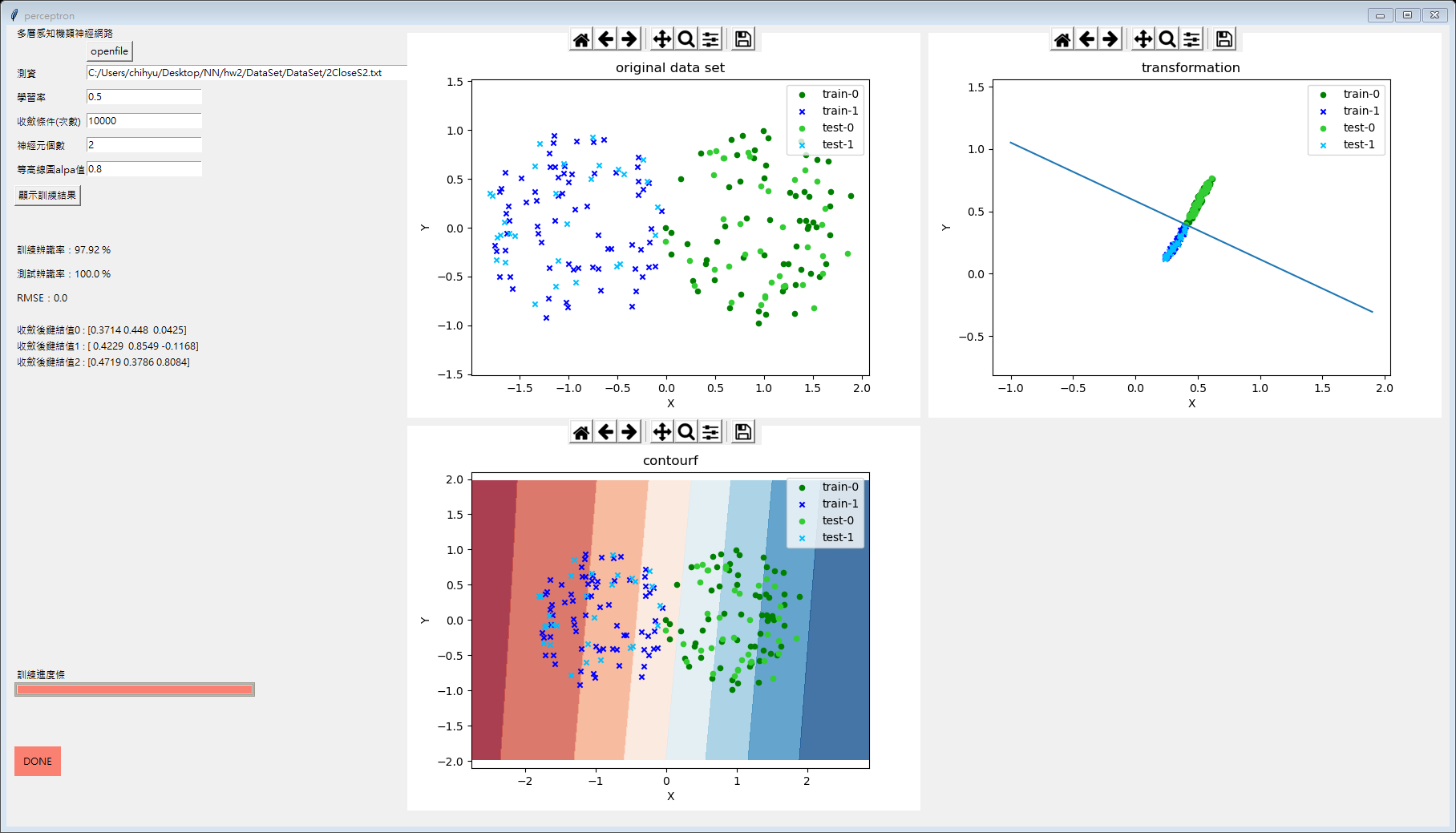
* 2Circle2：此測資為上一個測資加入後面雜點，程式判斷為雜點於是另外畫出。



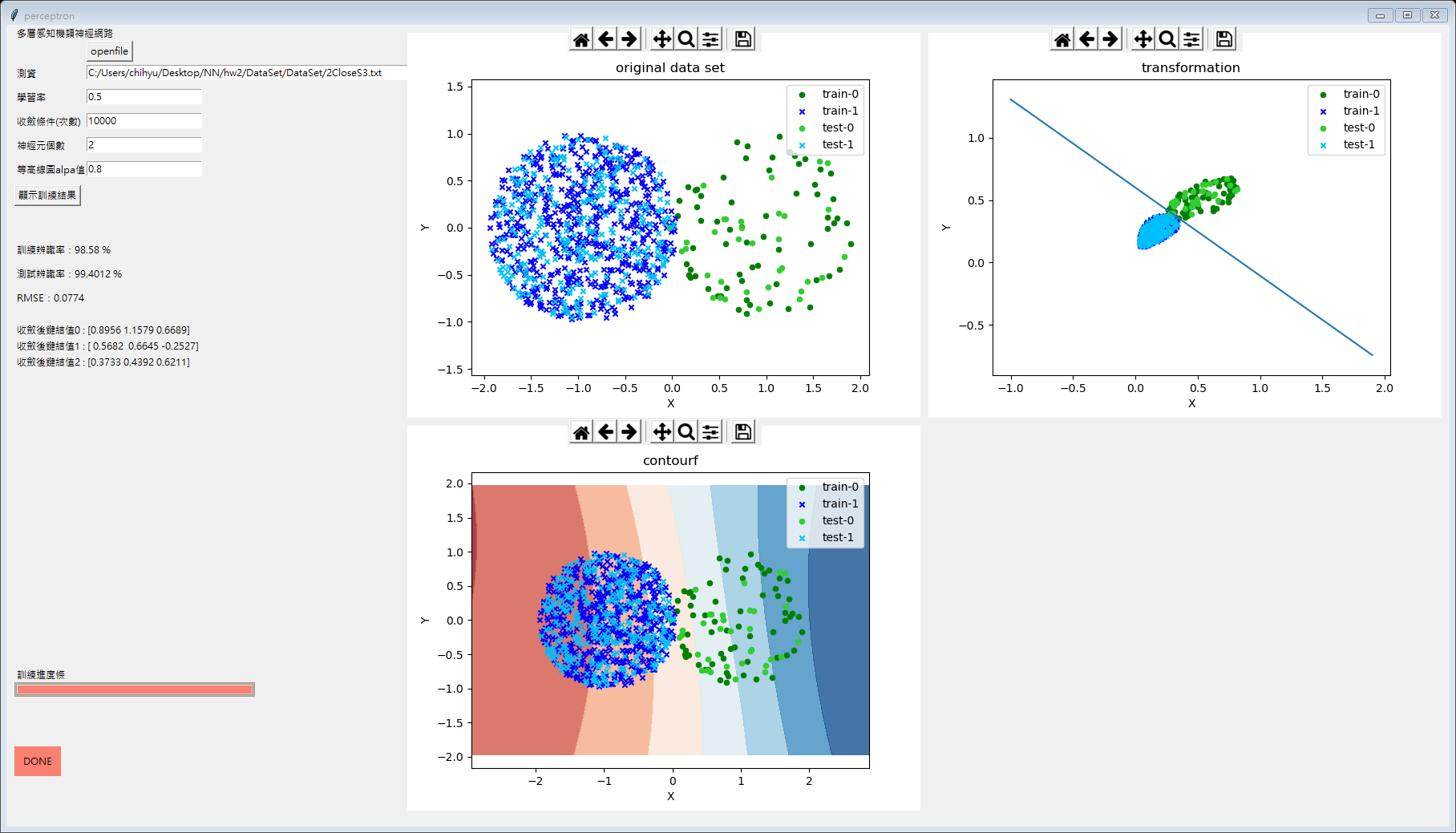
* 2CloseS：當收斂次數適當，測試辨識率即能100%。



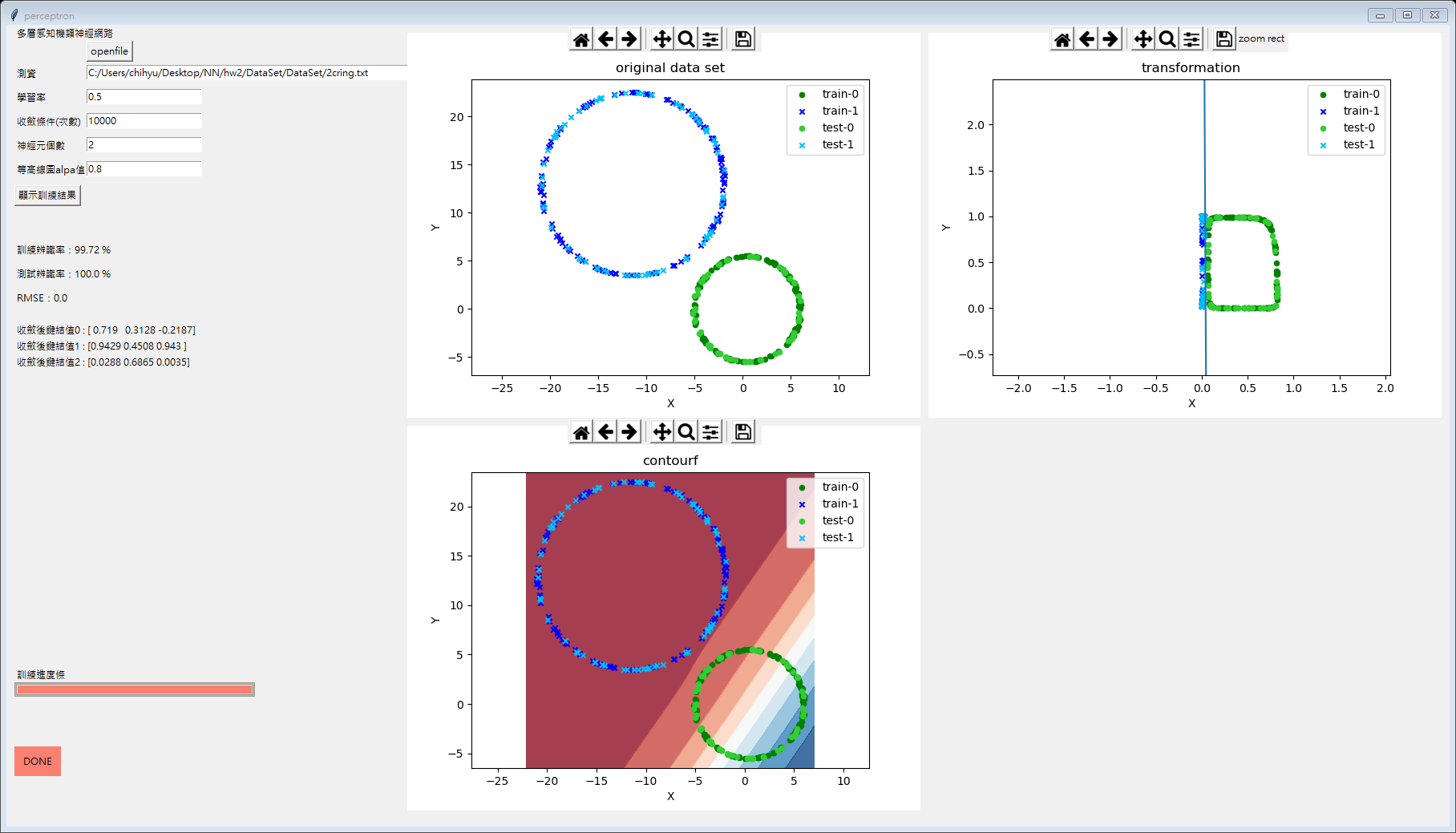
* 2CloseS2：這個測資相似2CloseS測資，中間地帶雖然較模糊，但透過多層的架構可以完善分出。



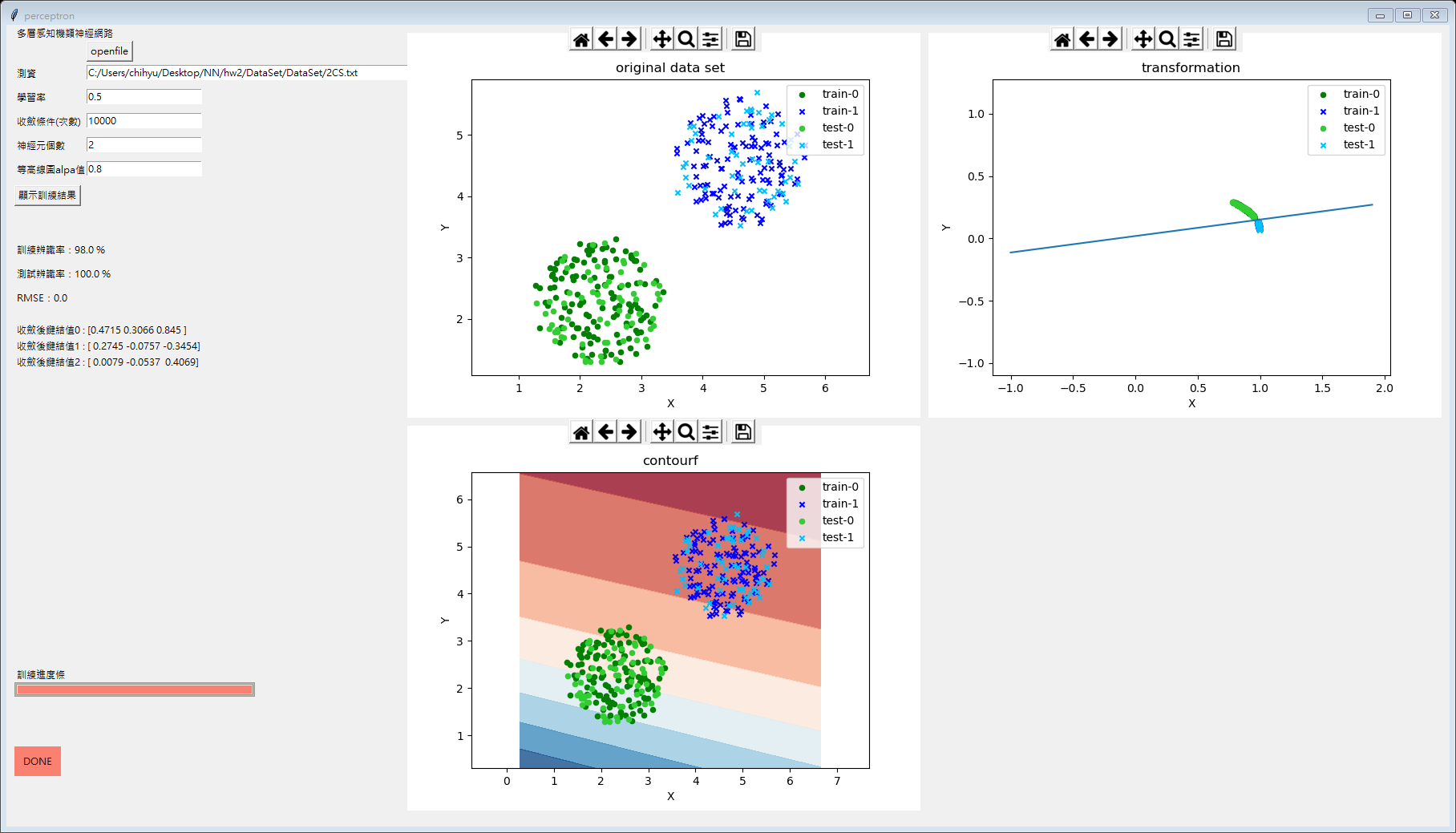
* 2CloseS3：這個測資也是當收斂次數適當，測試辨識率就會高。



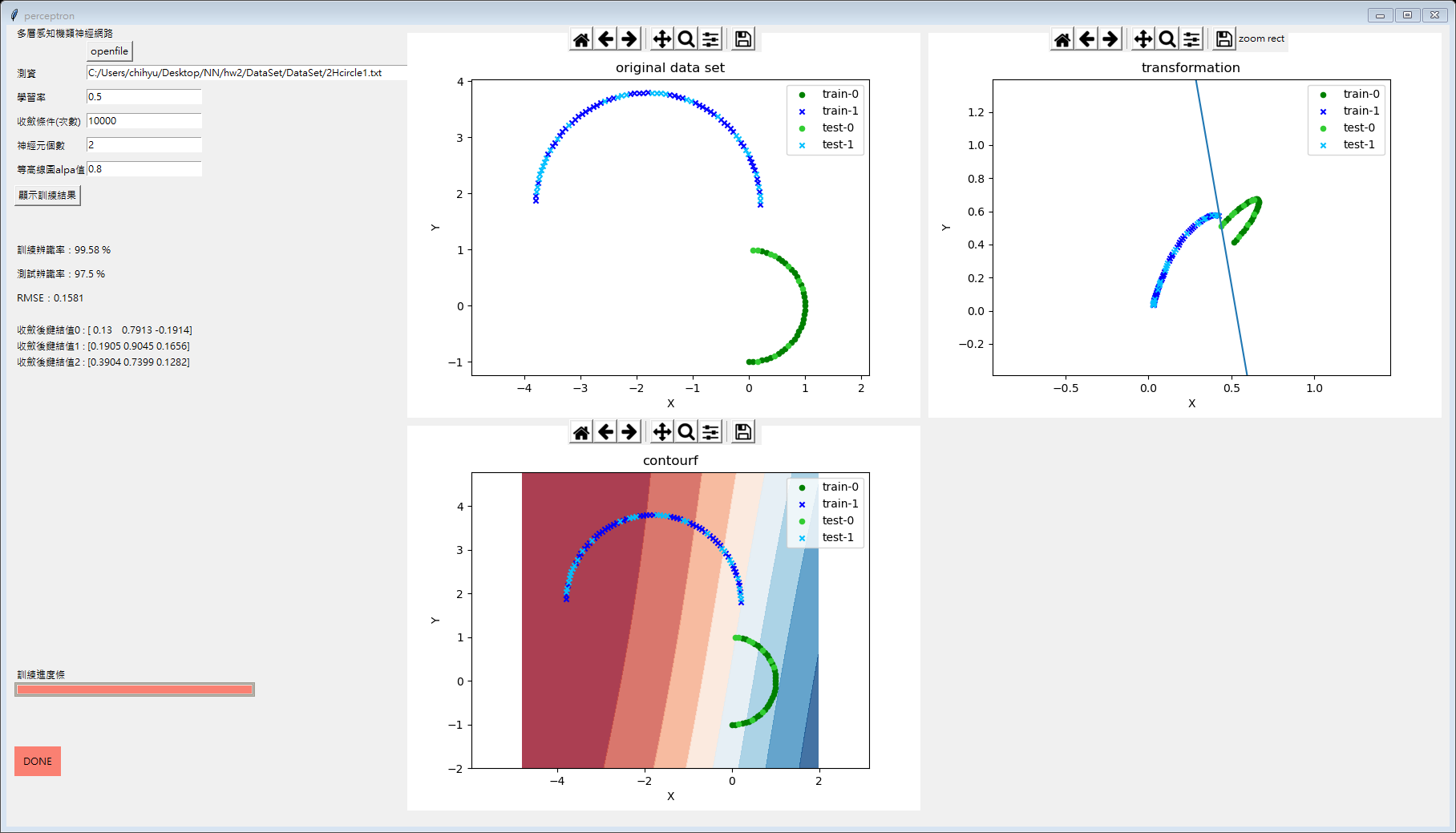
* 2cring：此測資無模糊地帶的data，當收斂次數適當，即可以100%辨識。



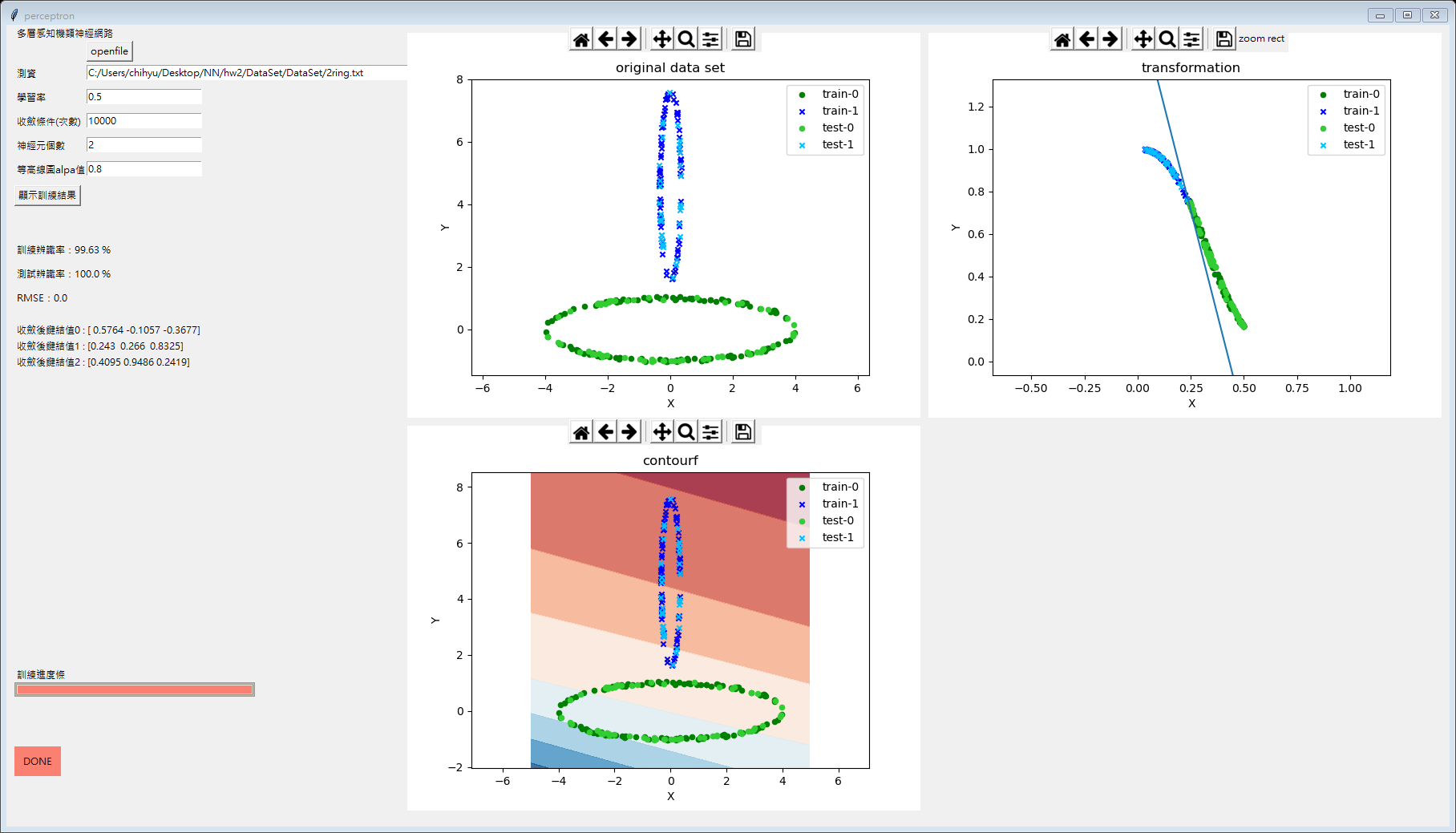
* 2CS：這個測資也是當收斂次數適當就能準確的辨識。



* 2Hcircle1：此測資也是經過適當收斂條件即能有良好辨識率。



* 2ring：這個測資經過適當的收斂次數就能準確辨識testing data。



1. 實驗結果分析及討論

XOR dataset因非線性分割且資料筆數少，所以如果想要準確分割必須將四筆資料都餵入訓練。如果分3：1或2：2都無法有良好的效果，我測試多神經元也無法良好的分割，所以我覺得是因為資料筆數少所以才無法完善訓練。因此我將四筆資料都餵進訓練，透過等高線圖可看出分割得很好，並測試準確率能100%。如果只餵三筆或兩筆資料訓練都會使得結果較像線性分割的例子。

鍵結值向量的初始值對網路的學習演算法是否收斂不會有影響，只會加快或減慢網路的收斂次數，所以我將鍵結值的初始值以隨機的方式產生。隱藏層中神經元個數，雖然我有實作設定個數，但我發現只要兩個神經元就可以有很好的效果，我認為是這幾個dataset並非複雜，而有此結論。

畫圖的部分除了把原始資料畫出，透過投影片教的特徵空間轉換，畫成可線性分割。並畫等高線圖更便於觀察，可以更清楚每一區塊分類的情形。

此次作業相較於第一次的單層感知機的實作，各資料集都顯示更好的測試率。

並透過等高線圖可看出多層感知機處理非線性資料集的效果。這次的作業我也對GUI的版面修改，使得更便於使用者操作及觀察，透過進度條的顯示，可以讓使用者知道訓練的速度並不會不知道要等多久，最後在所有程式包括畫圖完成後顯示”DONE”讓使用者知道程式已經全部正確跑完。

感知機的學習過程並不保證一次就學會，有時更會發生為了學新的輸入卻將原先以正確分類的資料給誤判。經過這次的作業可以更了解多層感知機運作理論，藉由倒傳遞演算法算出區域梯度函數，使得可以更好的修改鍵結值。

1. 加分項目：“隱藏層的神經元個數可設定”