Backpropagation

作者：吳乙澤

一、 網路架構說明

本專案利用倒傳遞神經網路(Backpropagation Neural Network)對數據進行分類，該網路層數共三層，分別為輸入層、隱藏層、輸出層，且隱藏層可依使用者意志隨意更改神經元個數。誤差以交叉熵(Cross Entropy)配合梯度下降法(Gradient Descent)，對模型的鍵結值(Weight)與偏差(Bias)進行更新。

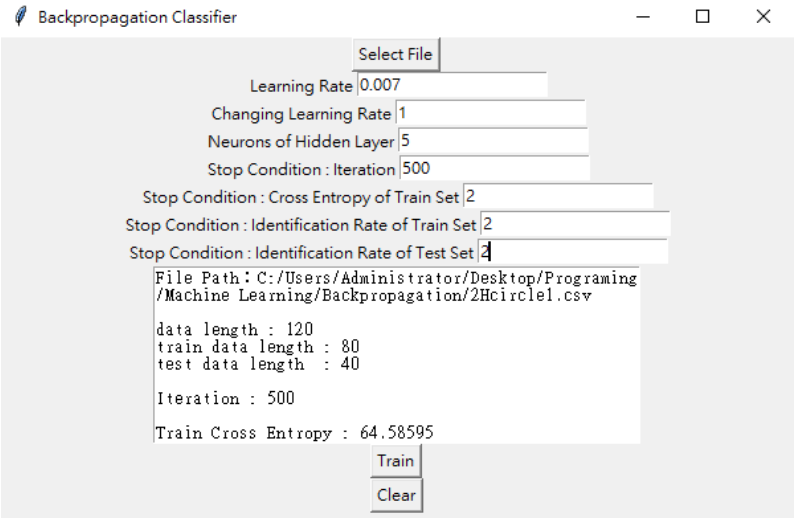
二、 程式碼簡介與說明

2-1 程式簡介

我的程式主要分成三個部分。第一是將 TXT 形式的資料轉成 CSV 檔，以利檔案的讀取。第二則是 Backpropagation 和圖形使用者介面(GUI)的演算法開發。其中，GUI 是用 Tkinter 進行設計的。第三是將 Backpropagation 的演算法和 GUI 程式打包成可執行檔(EXE)，以方便程式的開啟、資料集的訓練及後續的資料顯示。前兩個部份的程式為了開發方便，便在 jupyter notebook 的環境中進行開發。第三個程式由於打包需要 py 檔的緣故，因此我把 jupyter notebook 中 Backpropagation 的演算法和 GUI 程式碼複製到 spyder 中，再從 console window 裡用 pyinstaller 將 py 檔打包成 EXE 檔。

2-2 程式執行說明

將 Backpropagation_GUI.exe 點開，並稍等一會兒後，程式介面便會出現在畫面上。如下圖所示，按下 Select File，可以挑選 CSV 類型的二分類資料集進行訓練。Select File 下方有七個使用者自訂輸入，使用者可根據自我意志，隨意更改學習率、學習率隨著訓練次數的變化率(每 50 次更改一次)、隱藏層神經元個數及停止訓練的條件，例如訓練次數、訓練集的交叉熵大小、訓練集的辨識率、測試集的辨識率等等。在訓練過程中，若有任一條件達成，訓練便會停止，並把相關圖形與資訊顯示出來。



2-3 重點程式碼說明

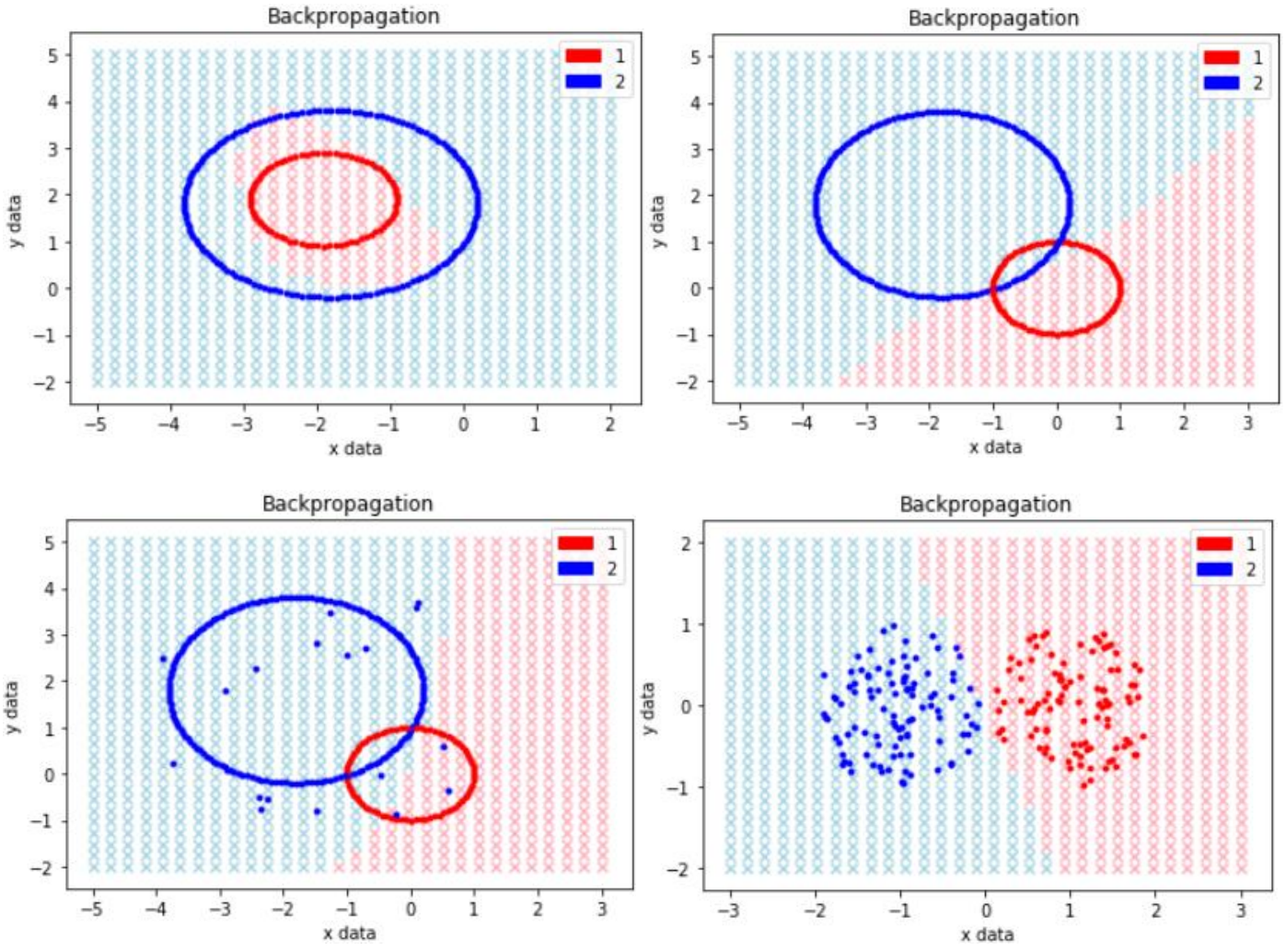
以下是 Backpropagation 計算更新值的程式碼。更新值計算的部分，由於 CSV 檔案中的標籤是 1

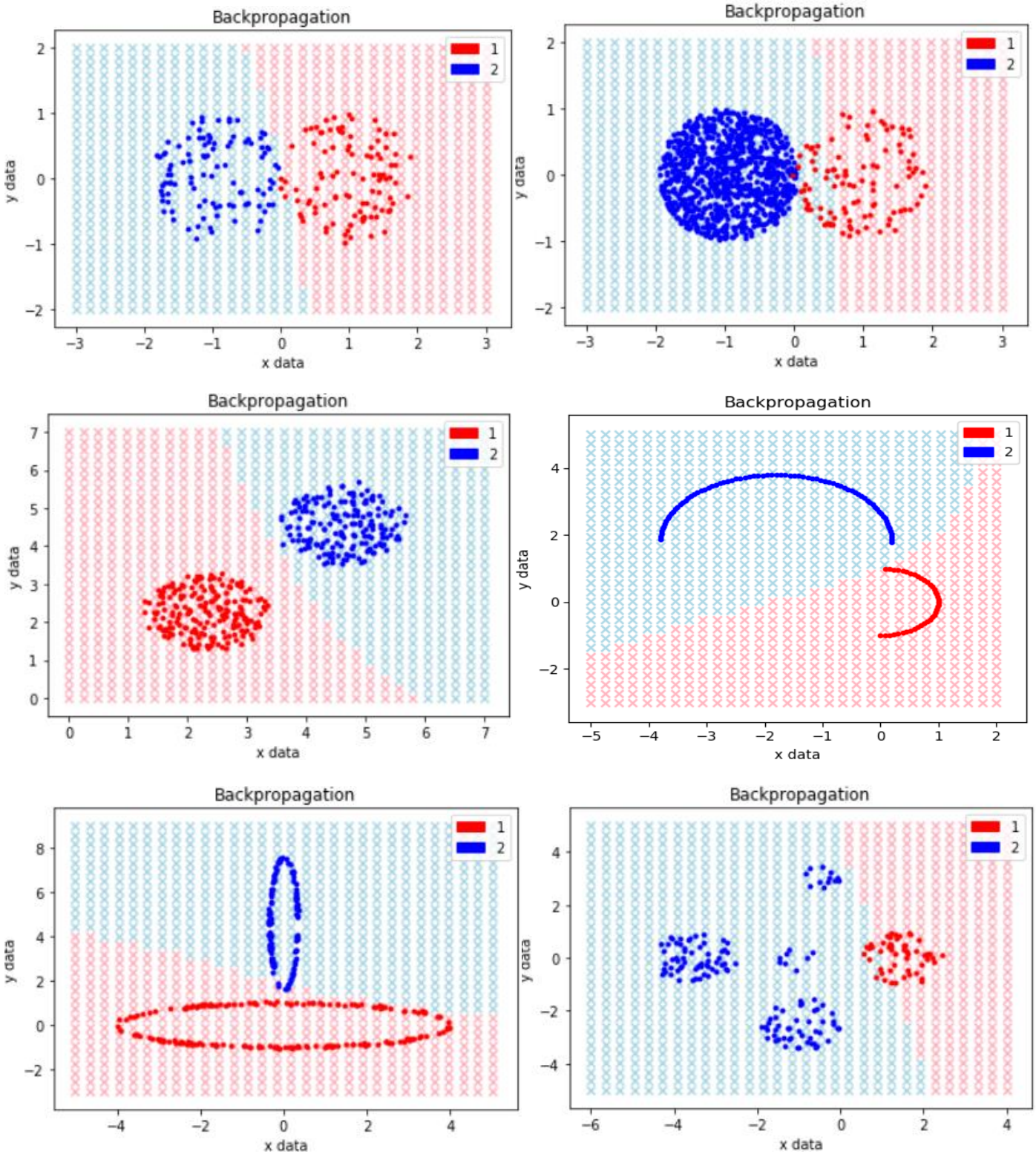
和 2，而 Cross Entropy 的公式需要 0 和 1 才能正常運作。於是，一開始我先將標籤值減 1，才進行後續的計算。其中，train_dict['2'][i]為標籤值，train_dict['0'][i]、train_dict['1'][i]則為不同的兩個輸入值。

```
# renew layer2
forwardPass = np.reshape(layerOutput[:NUM],(NUM,1))
layer2_changeValue[:NUM,:1] += forwardPass* train_predict[0]*(1-train_predict[0])* (train_predict[0]-label[0])
layer2_changeValue[:NUM,1:2] += forwardPass* train_predict[1]*(1-train_predict[1])* (train_predict[1]-label[1])
layer2_changeValue[NUM:NUM+1,:1] += train_predict[0]*(1-train_predict[0])*(train_predict[0]-label[0])
layer2_changeValue[NUM:NUM+1,1:2] += train_predict[1]*(1-train_predict[1])*(train_predict[1]-label[1])
# renew layer1
tmp = train_predict[0]*(1-train_predict[0])*(train_predict[0]-label[0])
tmp += train_predict[1]*(1-train_predict[1])*(train_predict[1]-label[1])
for i in range(NUM):
    layer1_changeValue[0,i] += train_dict['0'][i]* layerOutput[i]*(1-layerOutput[i])* tmp * layer2[i,0]
    layer1_changeValue[1,i] += train_dict['1'][i]* layerOutput[i]*(1-layerOutput[i])* tmp * layer2[i,1]
    layer1_changeValue[2,i] += 1* layerOutput[i]*(1-layerOutput[i])* tmp
```

三、 結果與討論

3-1 實驗結果





3-2 實驗結果分析與討論

見 3-1 實驗結果中的 10 張圖片，紅色表示類別一；藍色表示類別二；背景則以藍紅色的小叉表示該點被模型視為何種類別。上述的圖片進行訓練的時候，都有嘗試對遞迴次數、學習率、隱藏層神經元個數進行修改，以找到一個較好的參數達到更好的分類效果。另外，在訓練的過程中，我發現隱藏層的神經元數量越多，則分類結果越有非線性的傾向。尤其實驗結果中的第一張圖片，相比其他圖形而言，需要更高的非線性。因此在訓練的時候，我將模型隱藏層神經元數設至 40，並且重複訓練了幾次，才順利將該圖的整體辨識率拉到 0.958 左右。

綜觀整個實驗結果，該 Backpropagation 模型分類能力比 Logistic Regression 好，除了 2Circle2 資料集以外，全數圖片都有超過 90%，甚至 100%的分辨率。雖然 Backpropagation 比 Logistic Regression 的訓練速度慢了不少，但是 Backpropagation 具有 Logistic Regression 沒有的非線性分類能力，因此第一張圖片才能被分類出來。總的來說，神經網路在非線性問題上，會有比較好的結果。

SOURCE CODE : https://github.com/tailer954/Machine-Learning/tree/master/03_Backpropagation