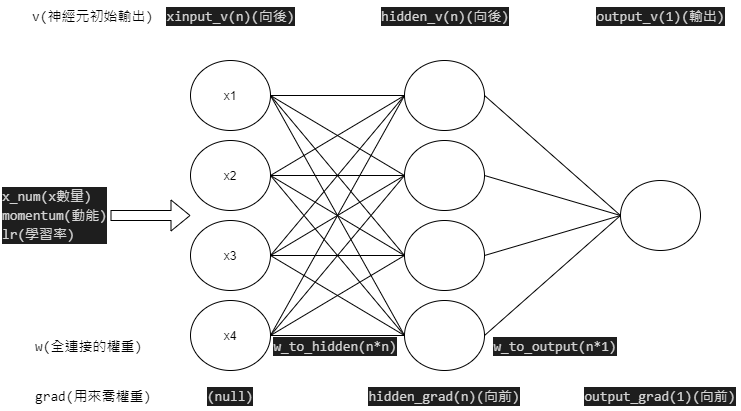
類神經網路 作業二\_多層感知機 大氣4A 黃展皇 106601015

工作環境：x64 windows10，conda 4.8.3，python3.6.10

1. 程式簡介，需包含實作架構（多層感知機or RBFN）

採用多層感知機的架構，如下圖：



程式碼簡介如下：

# Function set

# Calculate e(only for output layer)

def get\_error(d, y):

# Calculate sigmoidal(v) = y

def sigmoidal(v):

# Calculate sigmoidal'(v) = dy/dv

def sigmoidal\_diff(v):

# Calculate output layer gradient

def output\_gradient(d, v):

# Calculate hidden layer gradient

def hidden\_gradient(v, grad\_mul\_w\_sum):

# Calculate delta\_w(add momentum version)

def get\_delta\_w(momentum, w, lr, gradient, v):

# Neural network implementation

class Net():

# init net(n\*n\*1)

    def \_\_init\_\_(self, x\_num, momentum, lr):

        self.xinput\_v = np.zeros([x\_num])

        self.hidden\_v = np.zeros([x\_num])

        self.output\_v = 0.0

        self.hidden\_grad = np.zeros([x\_num])

        self.output\_grad = 0.0

        self.w\_to\_hidden = np.random.random([x\_num, x\_num])

        self.w\_to\_output = np.random.random([x\_num])

        self.x\_num = x\_num

        self.momentum = momentum

        self.lr = lr

# Check x shape,if ok then call forward and backward and fit\_weight

    def doit\_just\_doit(self, x, y):

# Calculate every node v

    def forward(self, x):

# Calculate every node gradient

    def backward(self, y):

# Fix every weight

    def fit\_weight(self):

# Calculate Eav

    def get\_eav(self, x\_dataset, y\_dataset):

# Calculate recognition rate

    def get\_recognition\_rate(self, x\_dataset, y\_dataset):

# Nonlinear trans(input x -> hidden y)

    def get\_nonlinear\_transformer(self, x\_dataset):

# Input dataset, randomly split to X and label \* training(2/3) and testing(1/3)

def split\_dataset(dataset):

# Plot point and weight line

def plot(dataset, w, X\_train, label\_train, X\_test, label\_test, file\_name\_we\_want, if\_nonlinear):

# Input weight w and point x, return y (w1x+w2y+w0=0 --> y=-(w0w1x)/w2)

def linear\_equations(w, x):

    return -(w[1]\*x - w[0])/w[2]

# Run when button be pushed.

def main():

執行資料集讀取、資料集解析分割、基礎值設定、資料集規一化與閾值加入、調用Net class並以train資料訓練網路，每100 epochs印出結果並每一epoch檢查結束要件(>最高訓練次數 or Eav不再進步 or 對測試資料集已能達到極高準確度)

if \_\_name\_\_ == '\_\_main\_\_':

執行tkinter gui介面，按按鈕執行main()

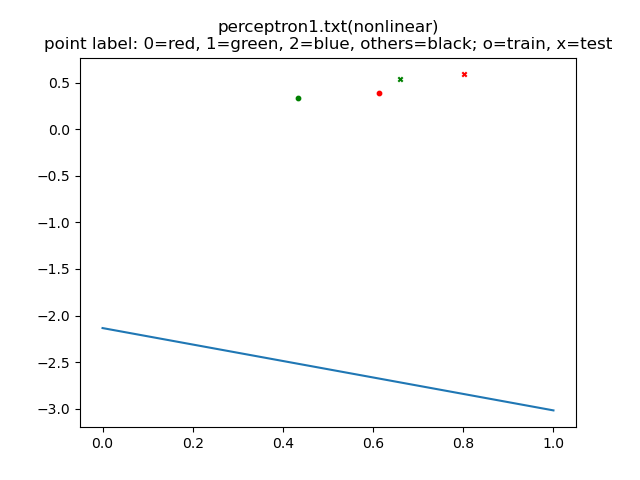
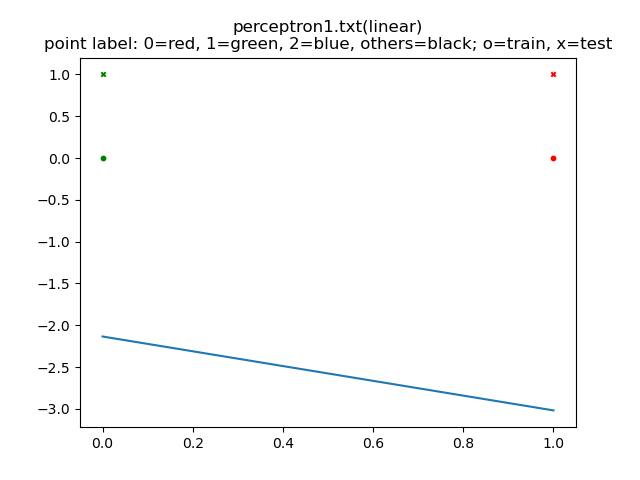
1. 程式執行說明（如何操作、使用）

執行附檔.exe檔後(勿移動資料集位置)可呈現一圖形化介面，可輸入學習率、欲執行資料檔案名稱、最大執行次數，經過計算後分別呈現線性/非線性2維分群結果圖片(分割線方程可能有誤)、訓練與測試準確度、訓練與測試方均根誤差以及最終的鍵結值。

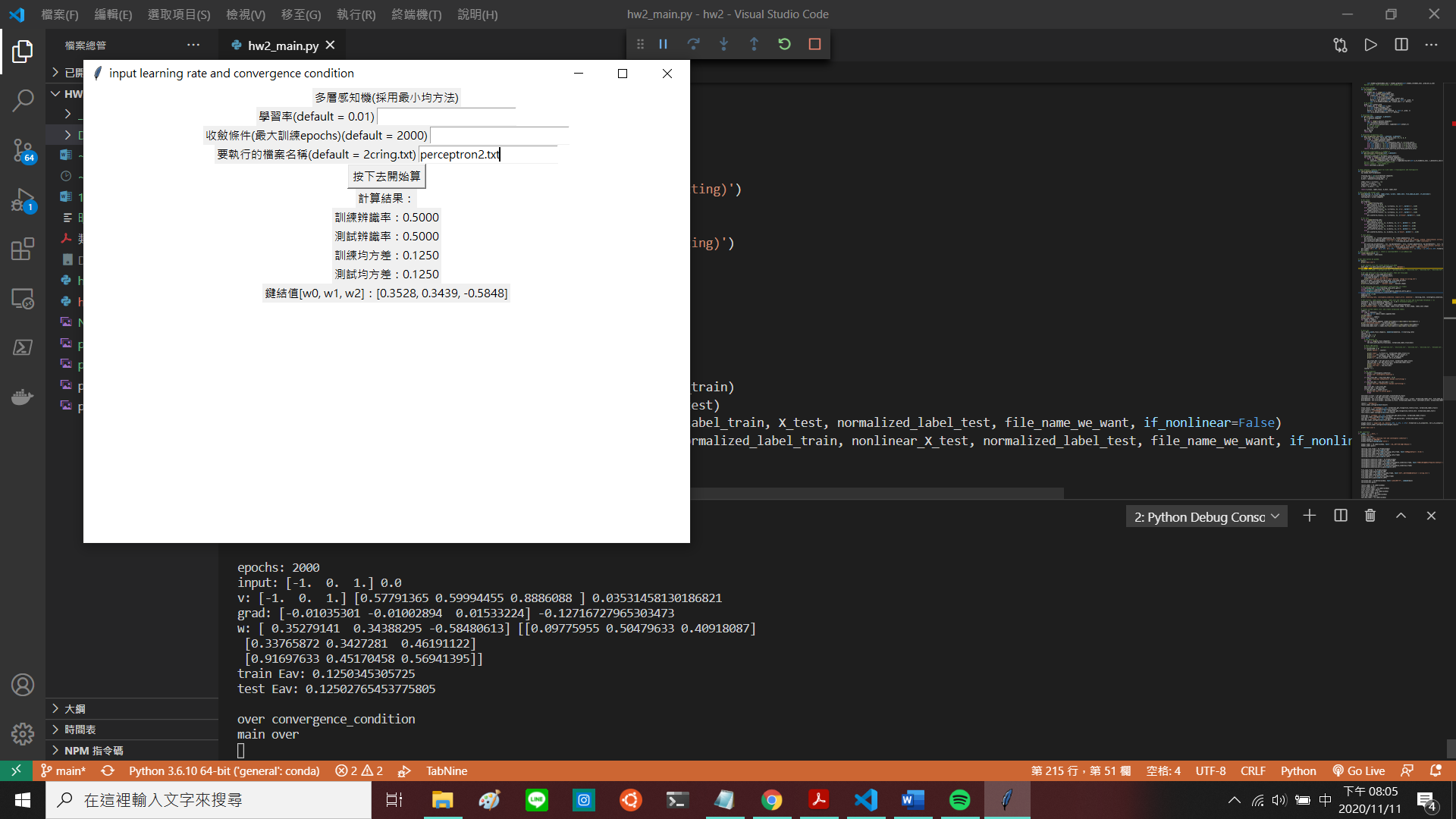
1. 實驗結果 (所有資料集都須有實驗結果和截圖及說明)

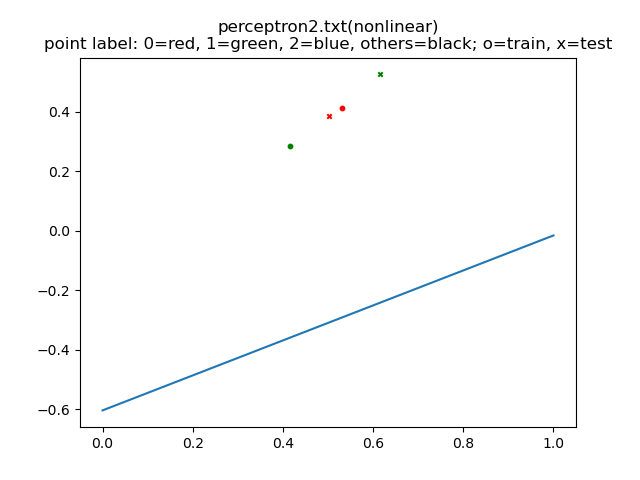
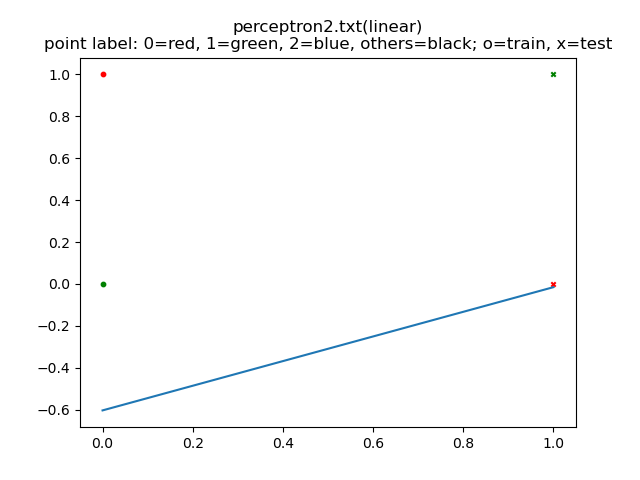
perceptron1.txt



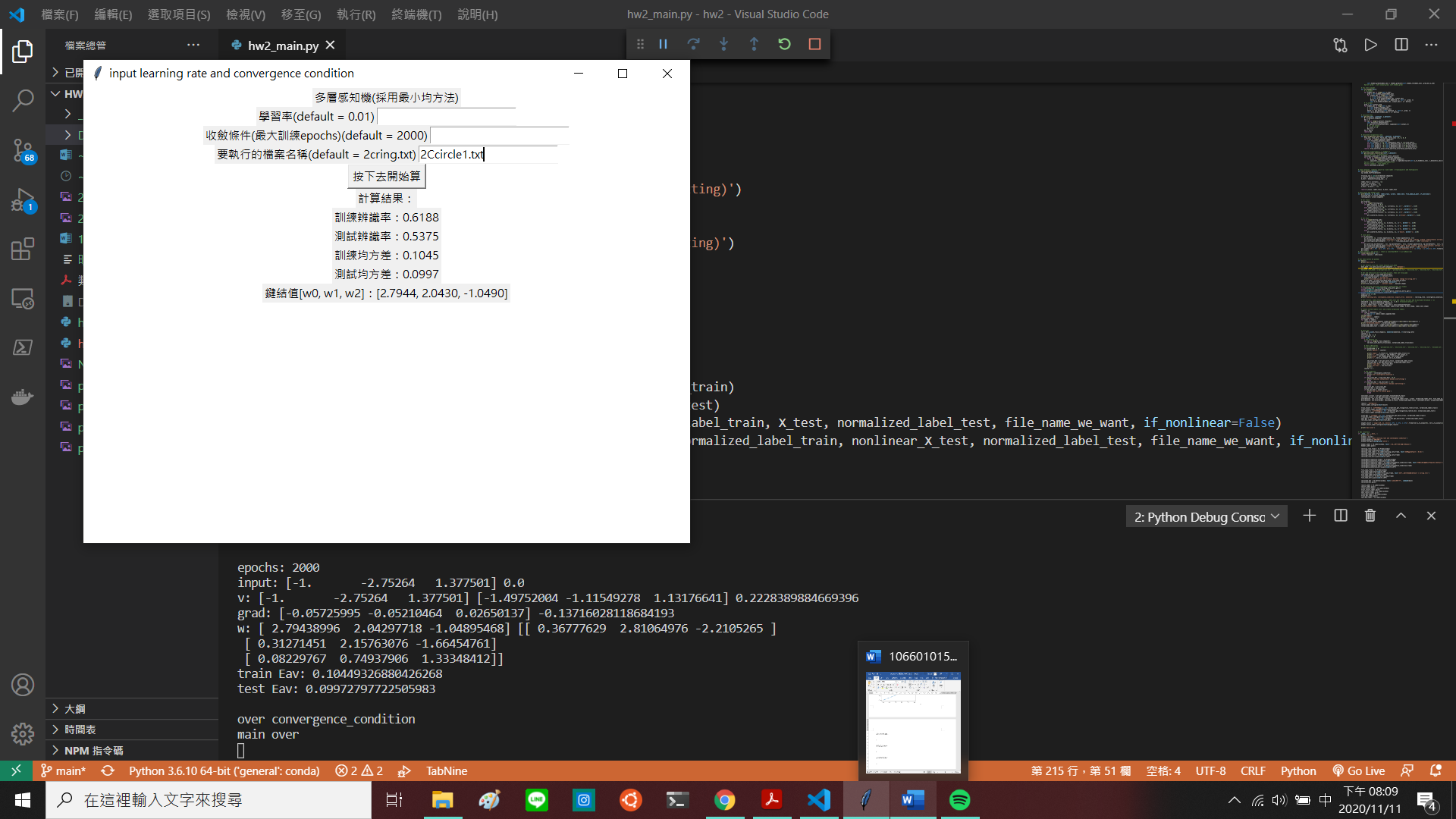


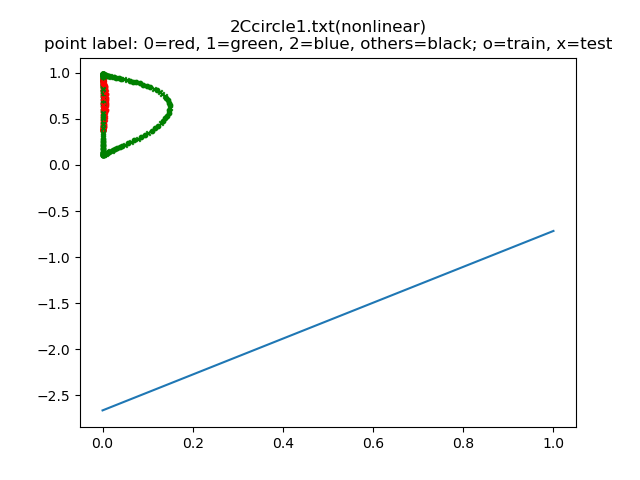
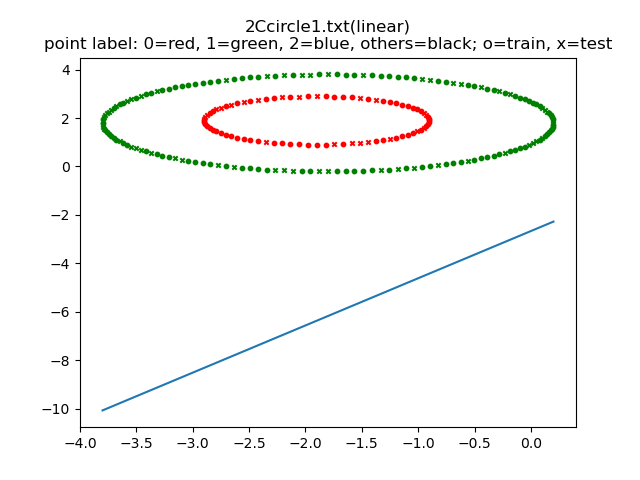
perceptron2.txt



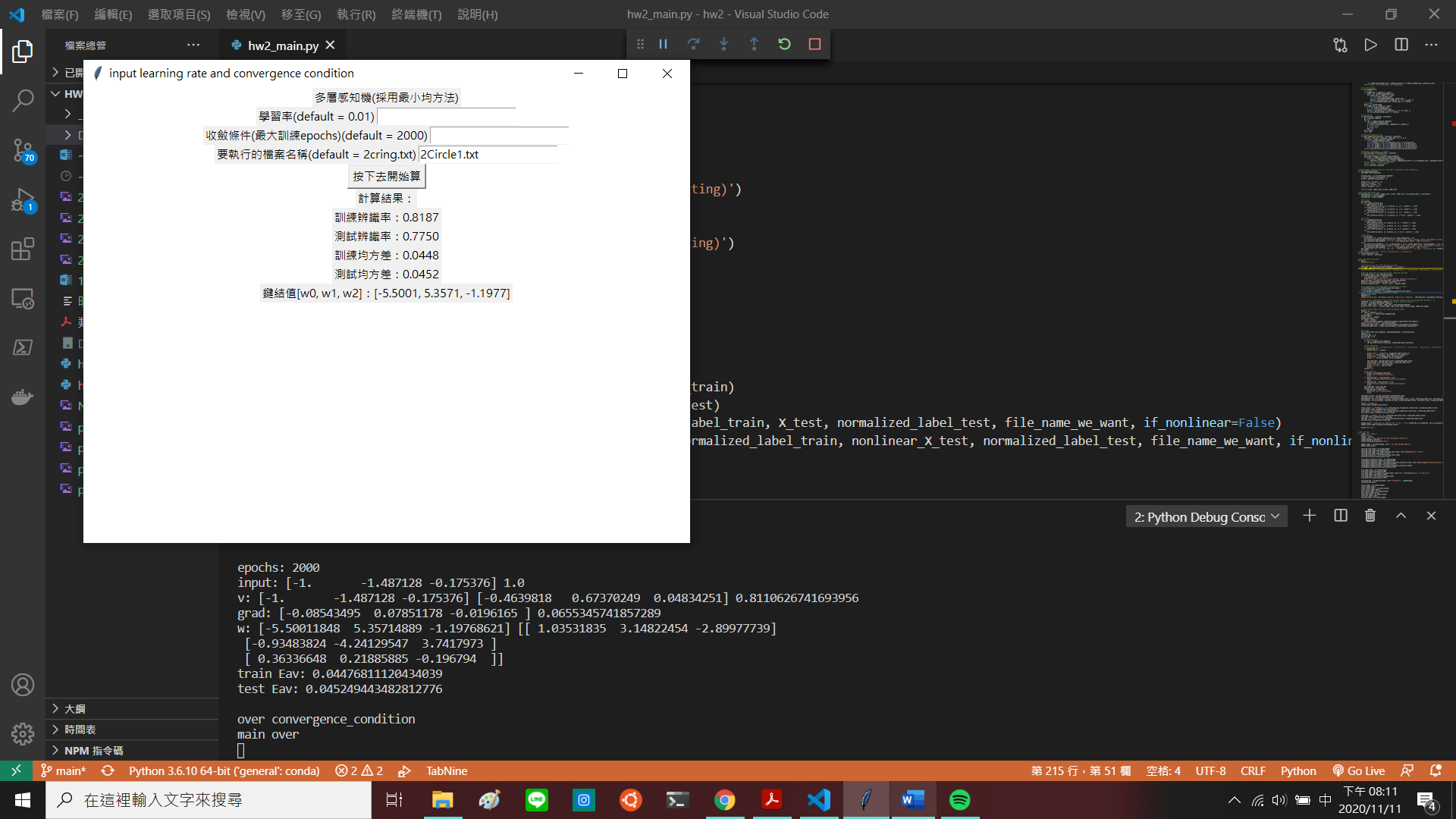


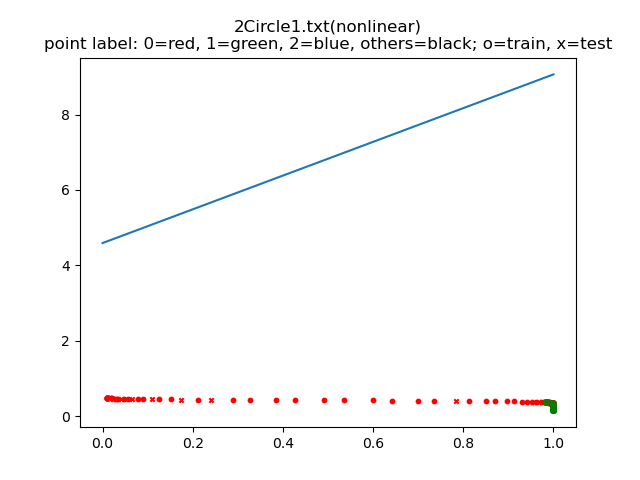
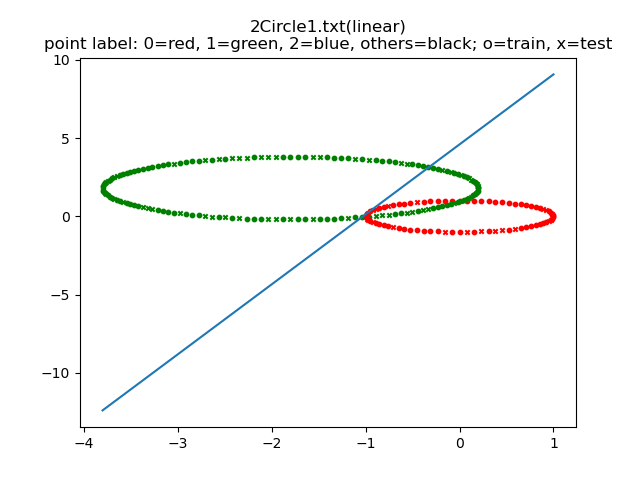
2Ccircle1.txt



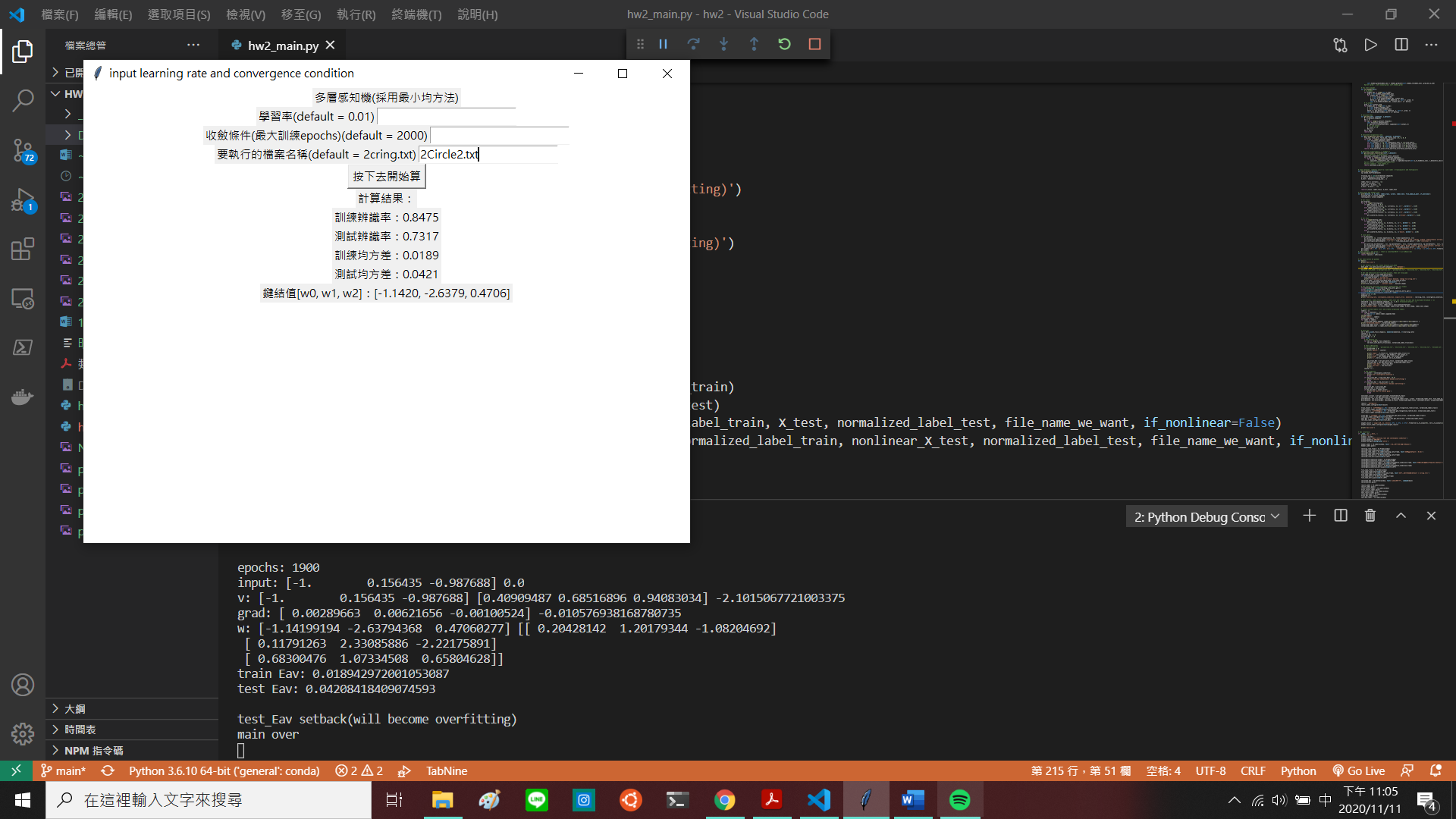


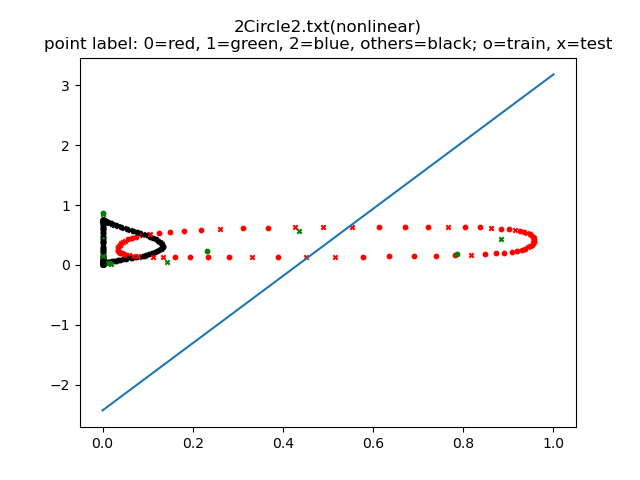
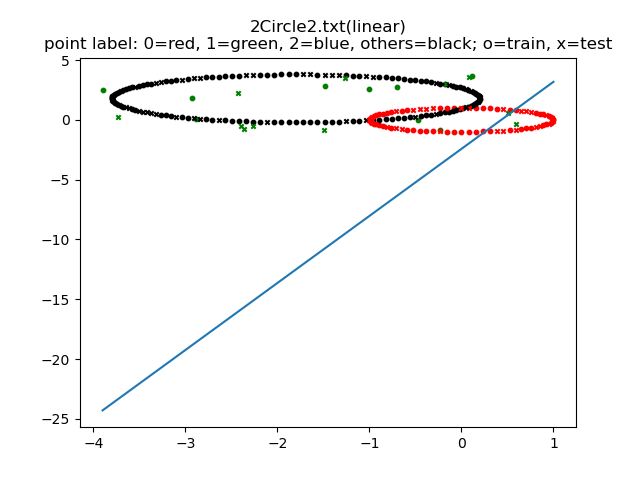
2Circle1.txt



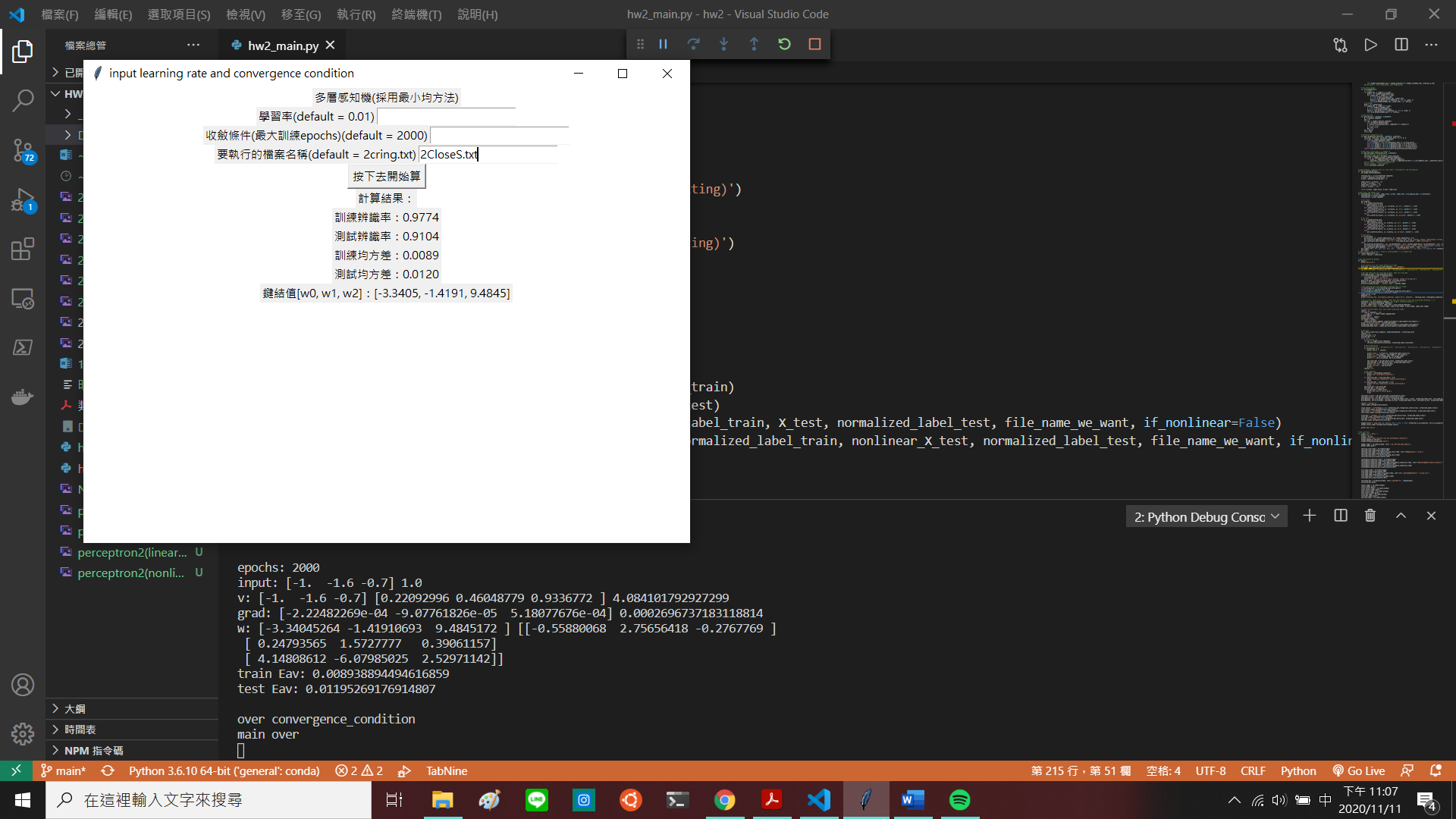


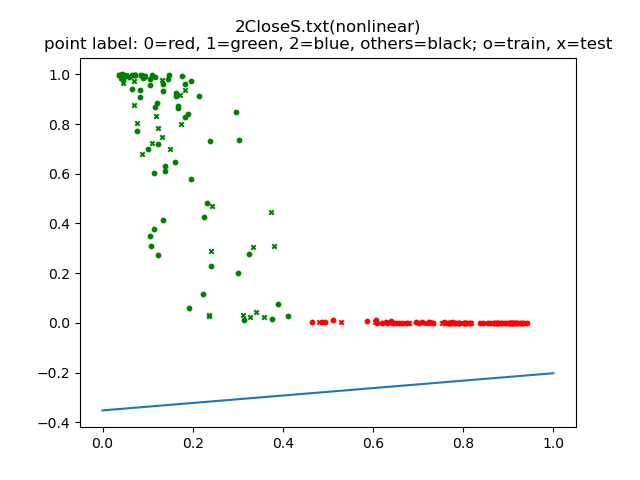
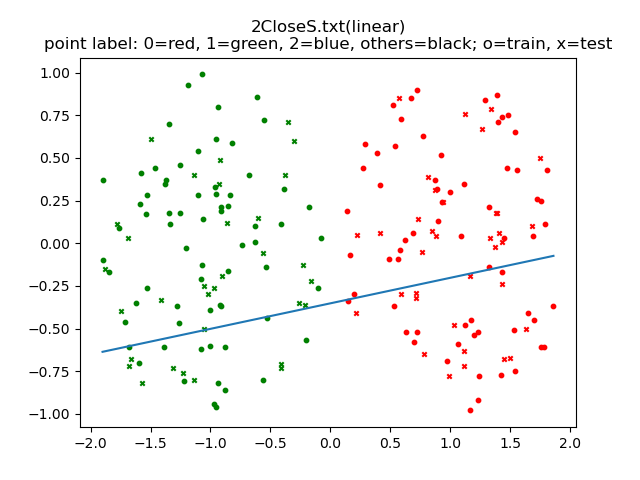
2Circle2.txt



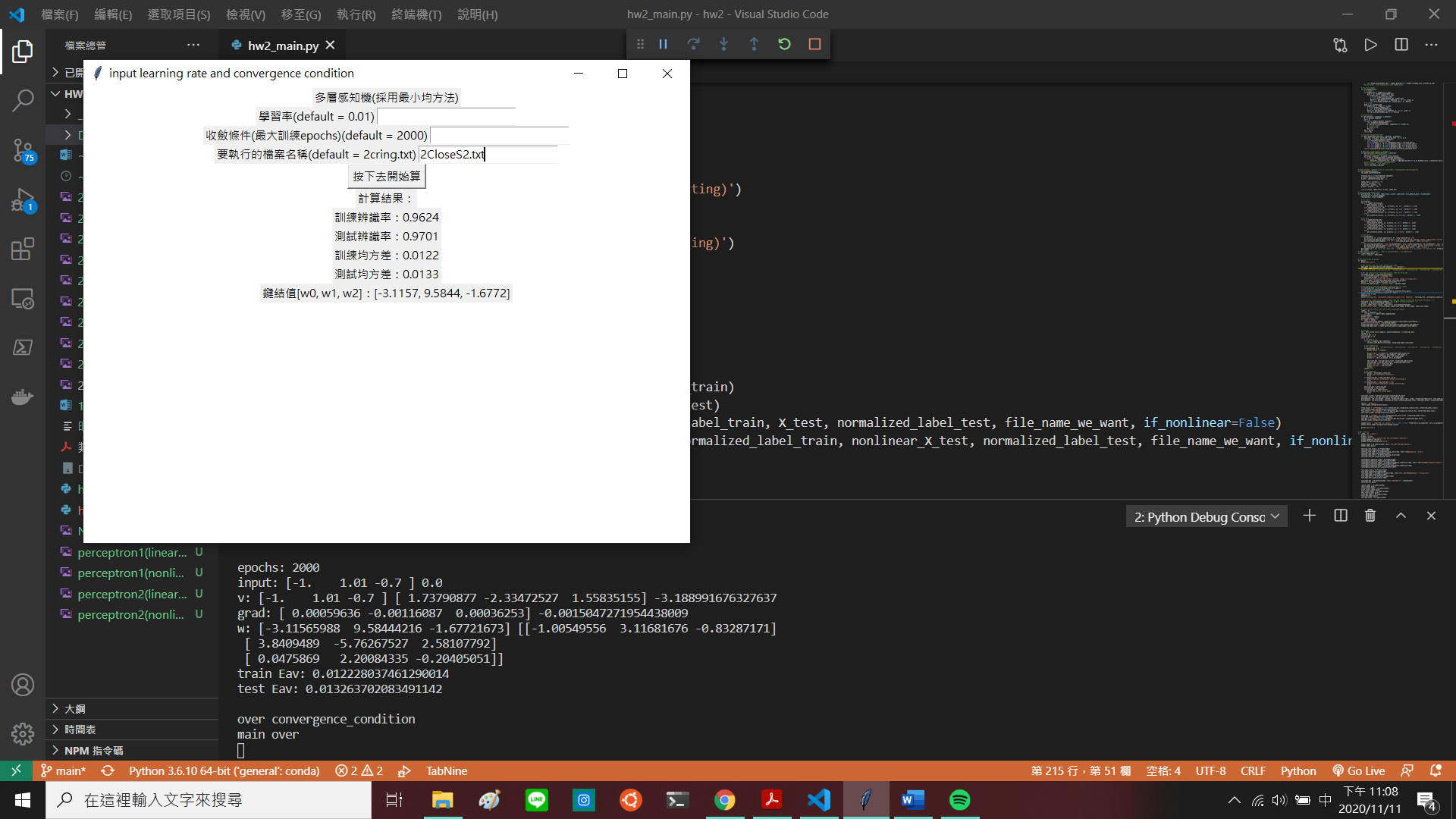


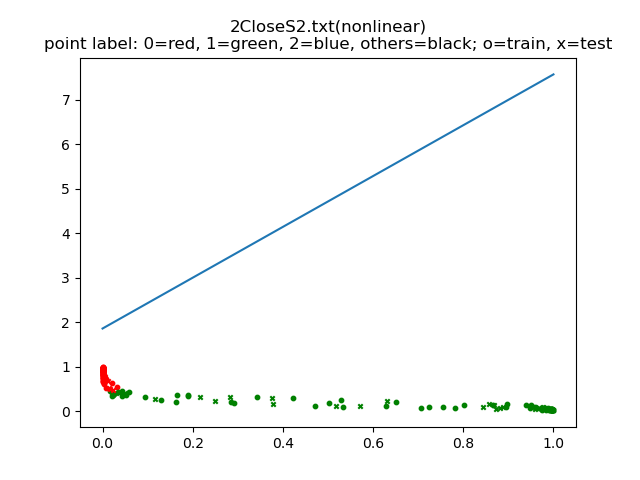
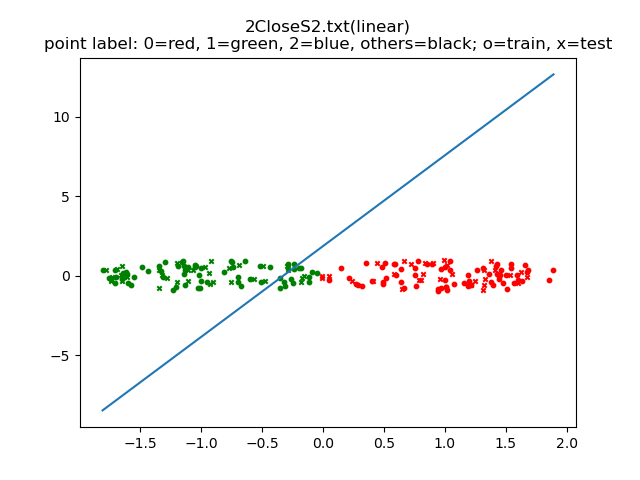
2CloseS.txt



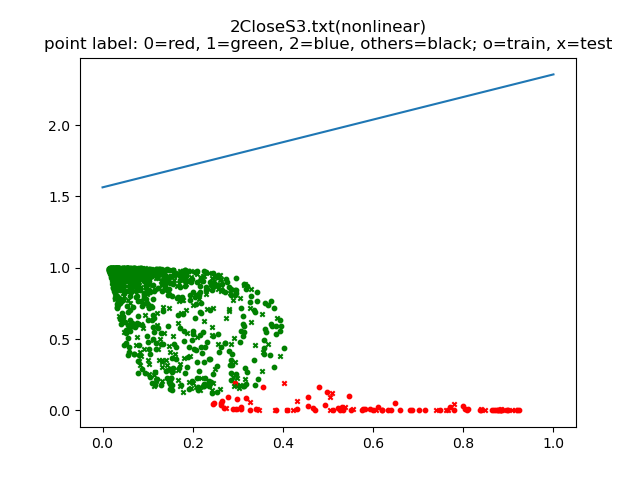
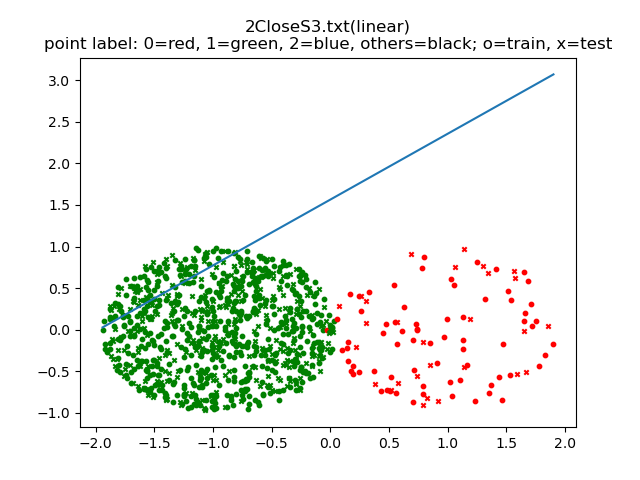
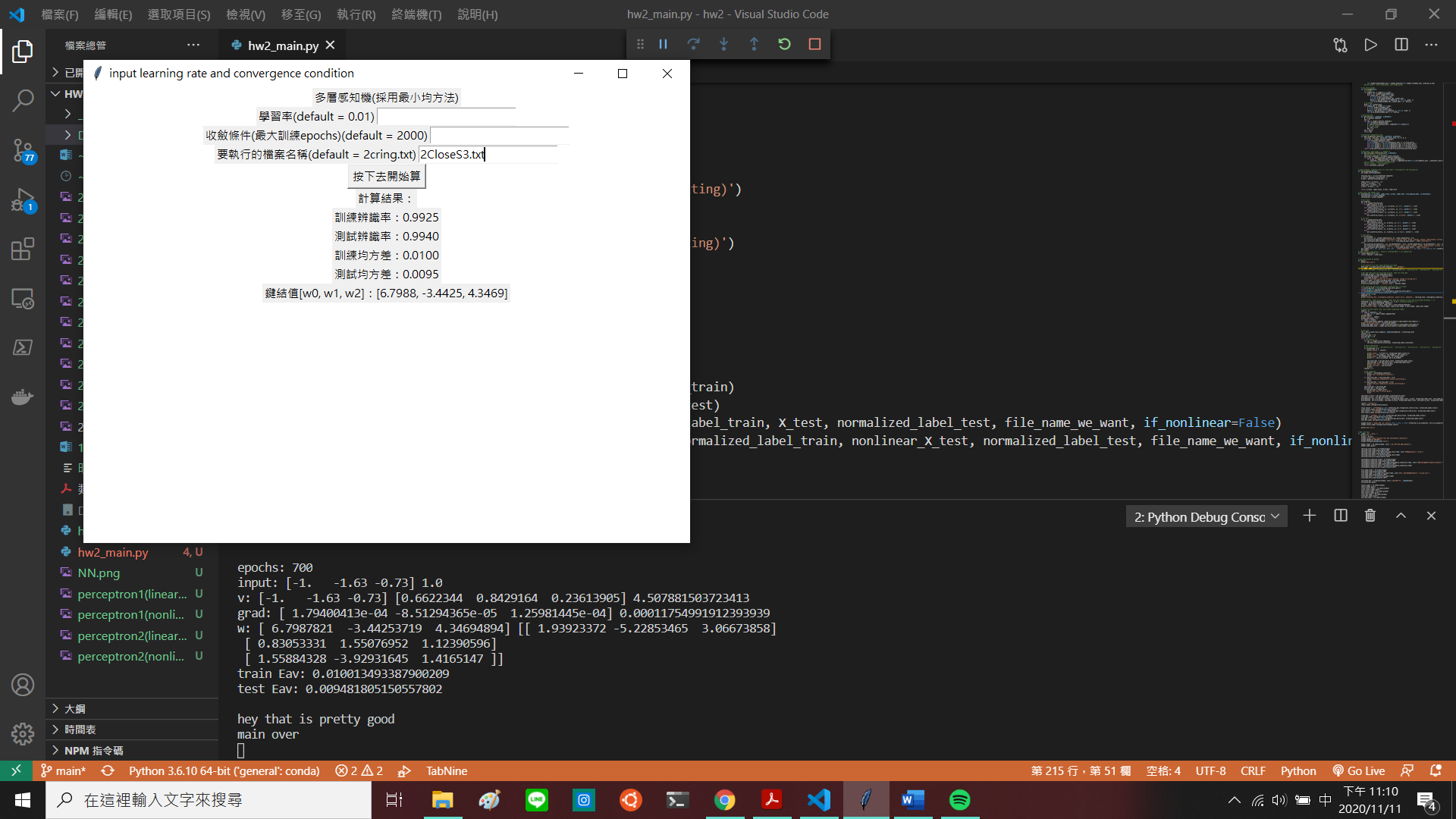


2CloseS2.txt

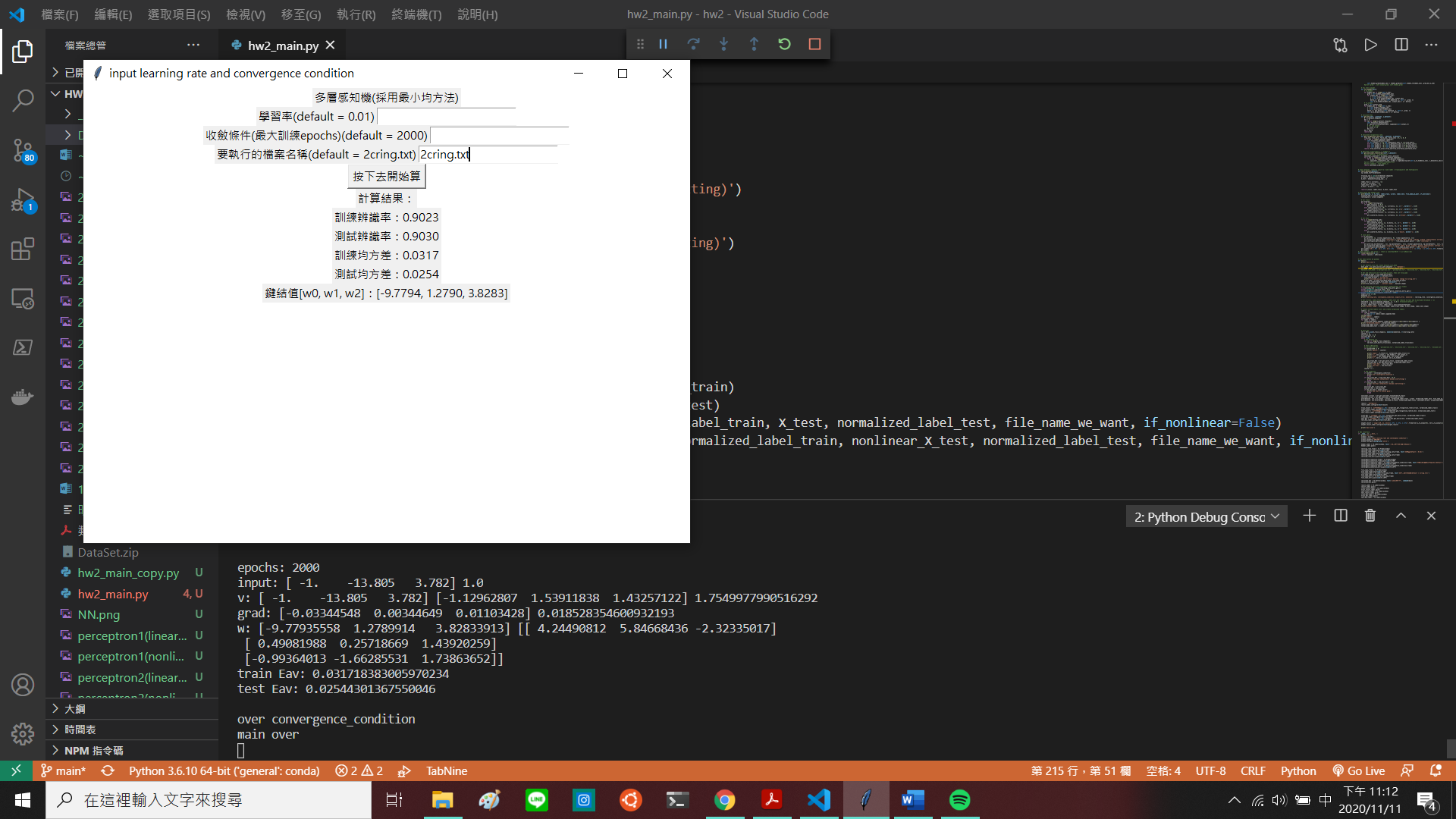


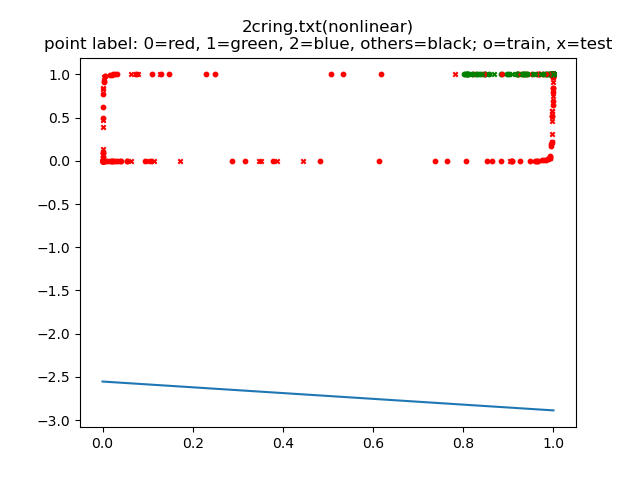
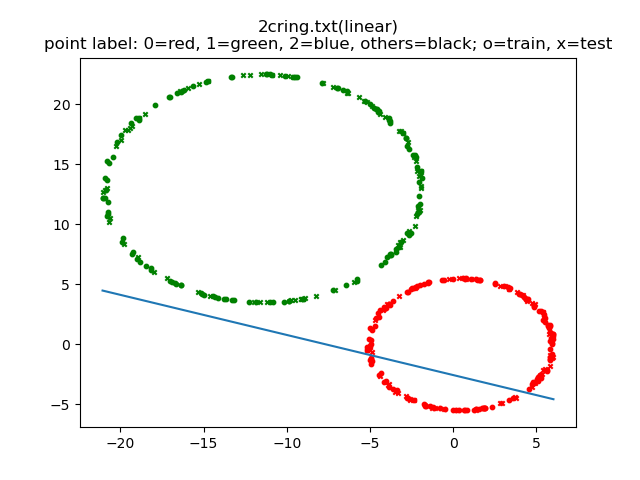


2CloseS3.txt

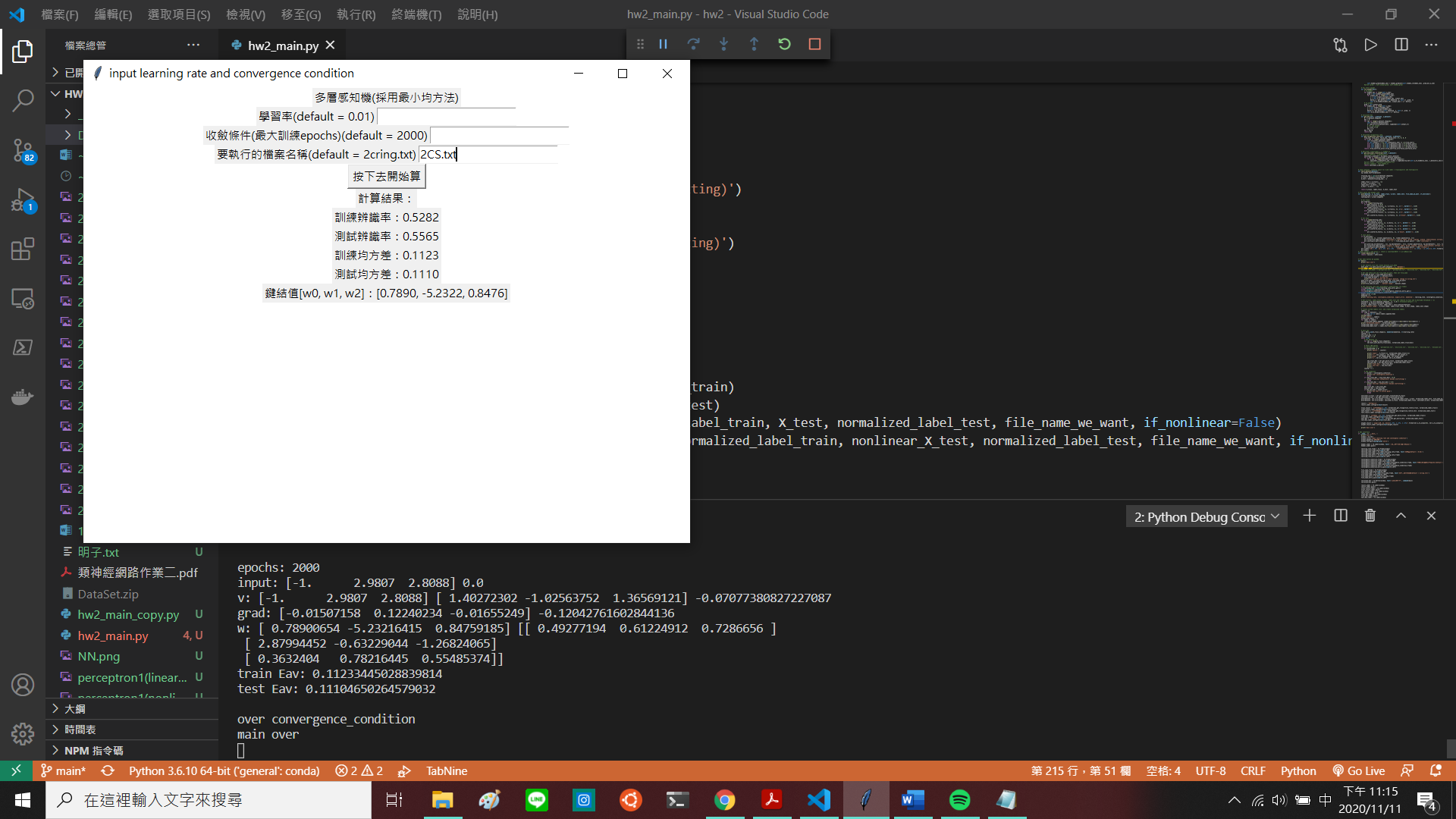


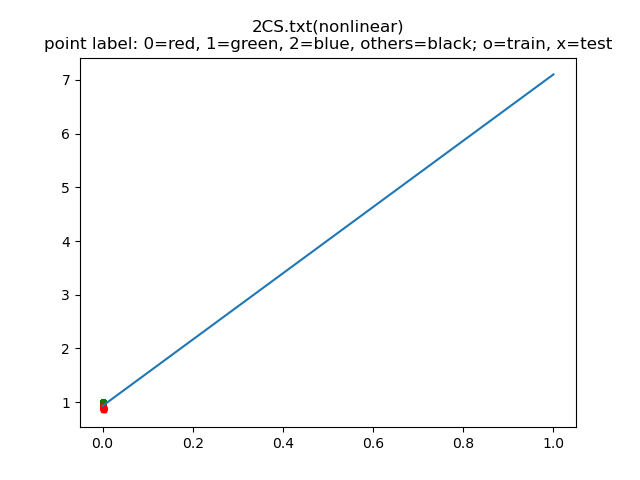
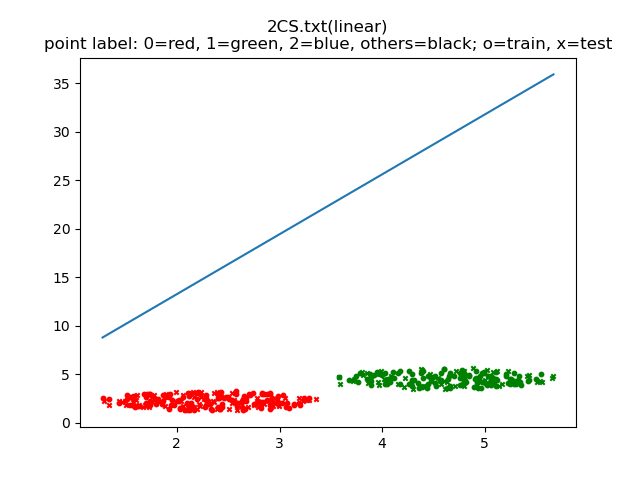
2cring.txt



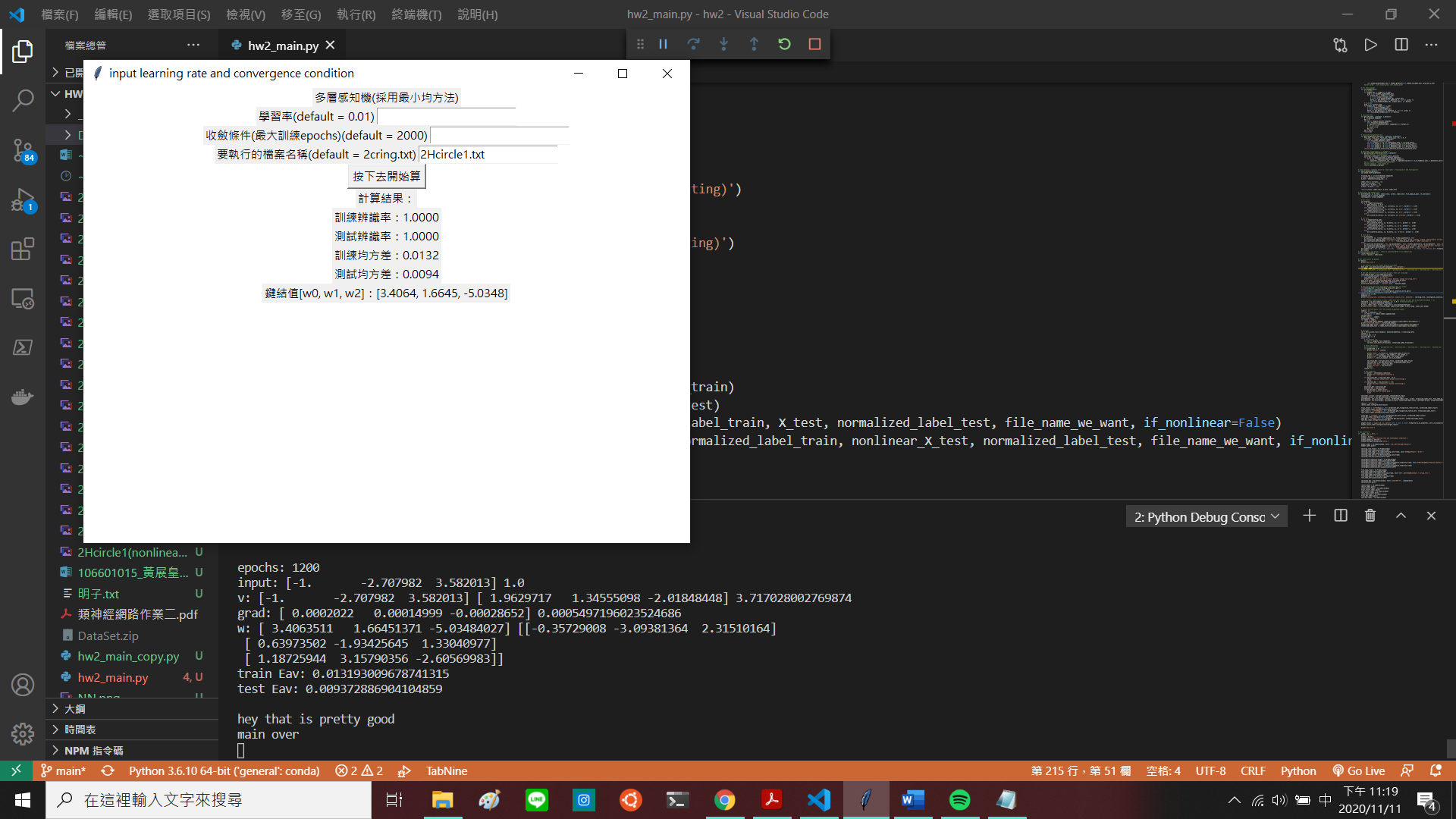


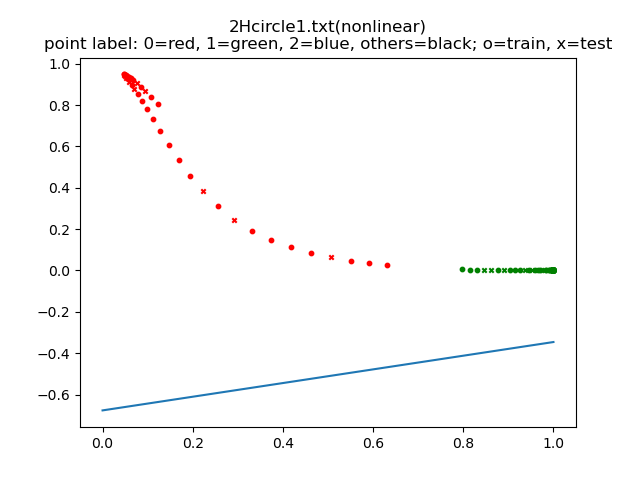
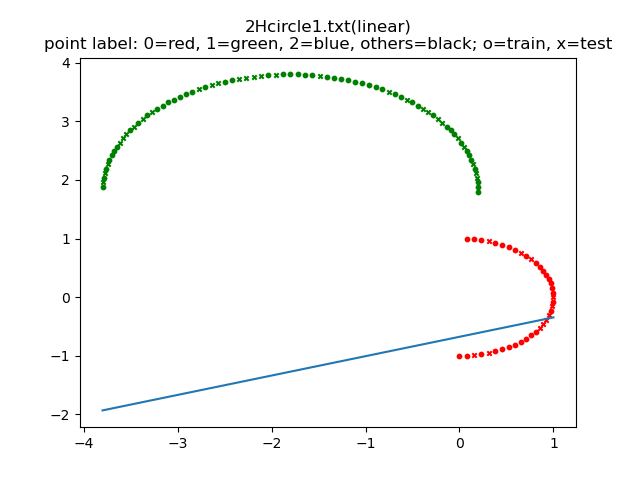
2CS.txt



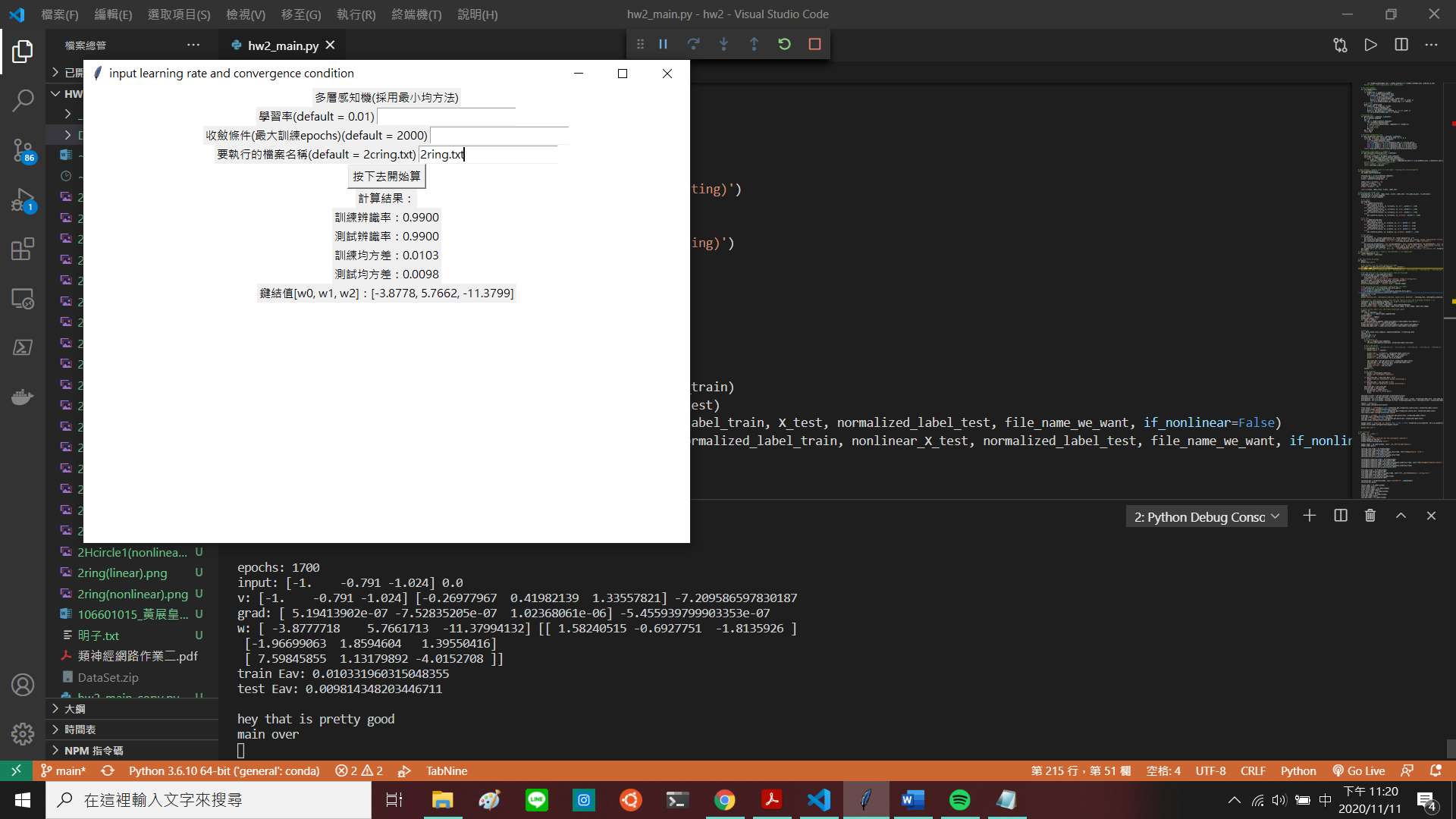


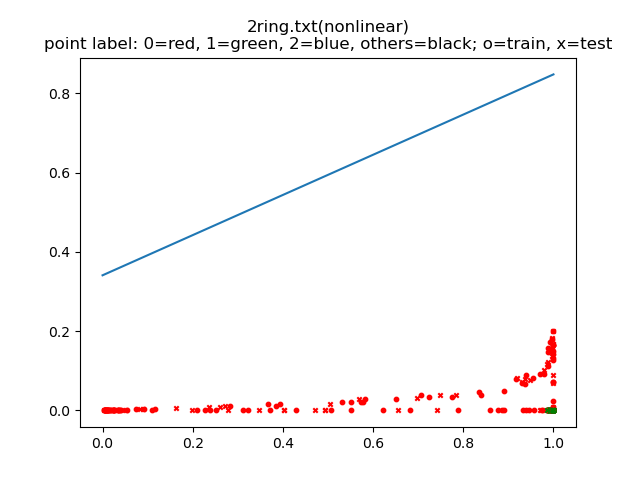
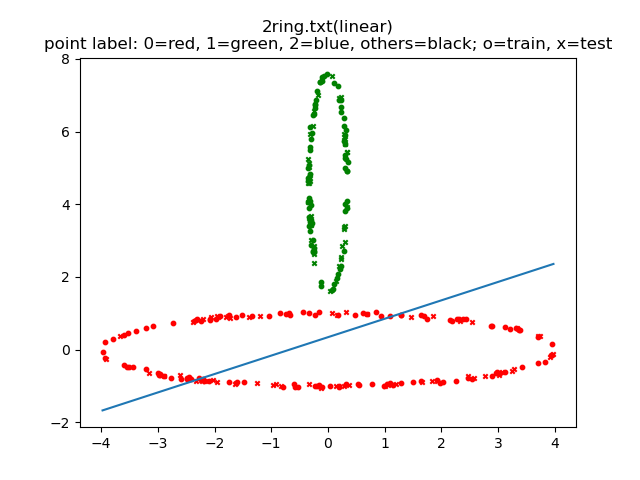
2Hcircle1.txt





2ring.txt





1. 實驗結果分析及討論。

非線性可分割資料集：perceptron2.txt、2Ccircle1.txt、2Circle1.txt、2Circle2.txt，其他均為可線性分割資料集。

如以上截圖的說明與資料呈現，基本上訓練出來的辨識率均有達到不錯的程度，非線性可分割資料集辨識率為[0.5, 0.6, 0.8, 0.8]，線性可分割資料集辨識率除了2CS.txt為0.5以外其餘均有0.9以上的成績。然而繪圖時遇到的瓶頸是不知道應該用哪個鍵結值作為線性方程的係數，以及係數背後的意義。因此分割線用隱藏層-輸出層最為線性方程權重，對線性/非線性資料集各做一次分割繪圖，然而視覺化效果仍然不佳。

考慮到無論是辨識率或是均方根誤差，本多層感知機在多個資料集均能有不錯的性能，因此應該是權重的線性方程式出現問題才導致繪圖出來的線並不能切割資料點。

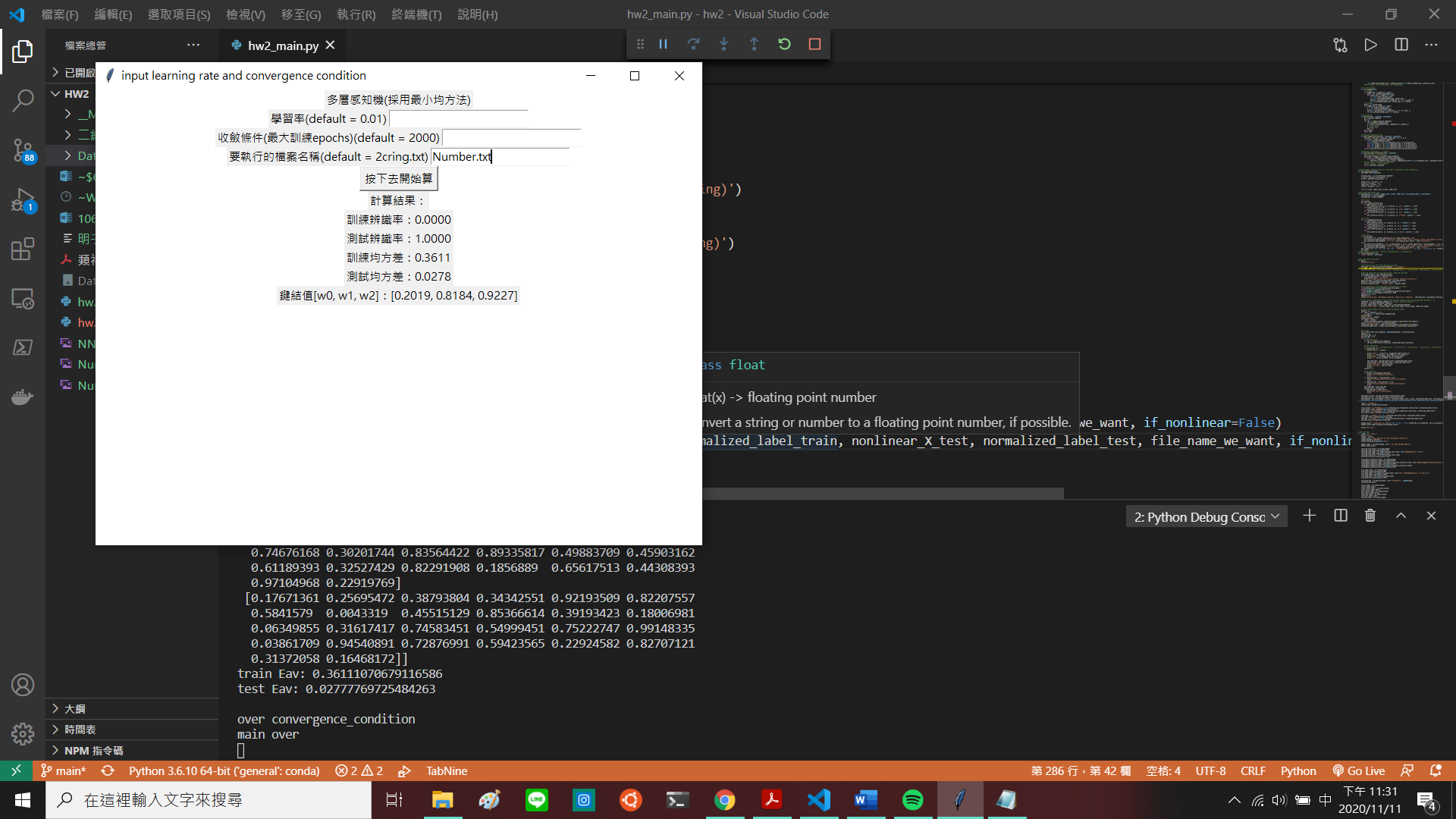
理論上本多層感知機可以解決所有資料集(train/test預測值趨近於100%)，然而可以看到在針對線性可分割的資料集仍然具有較高的辨識準確度。訓練次數的部分預設值為2000次，對於線性可分割資料集基本上都已經能達到很好的訓練效果，但是可能對於非線性可分割資料集還不夠，準確度仍然不夠低。

學習率預設值為0.01，在2000次的訓練下為適合的尺度，並且在過程中不會有所改變，泛化資料後可考慮變動學習率，即初始學習率稍大，而後每經過固定次數的訓練即降低學習率，不過目前固定學習率成果相當良好故不考慮。

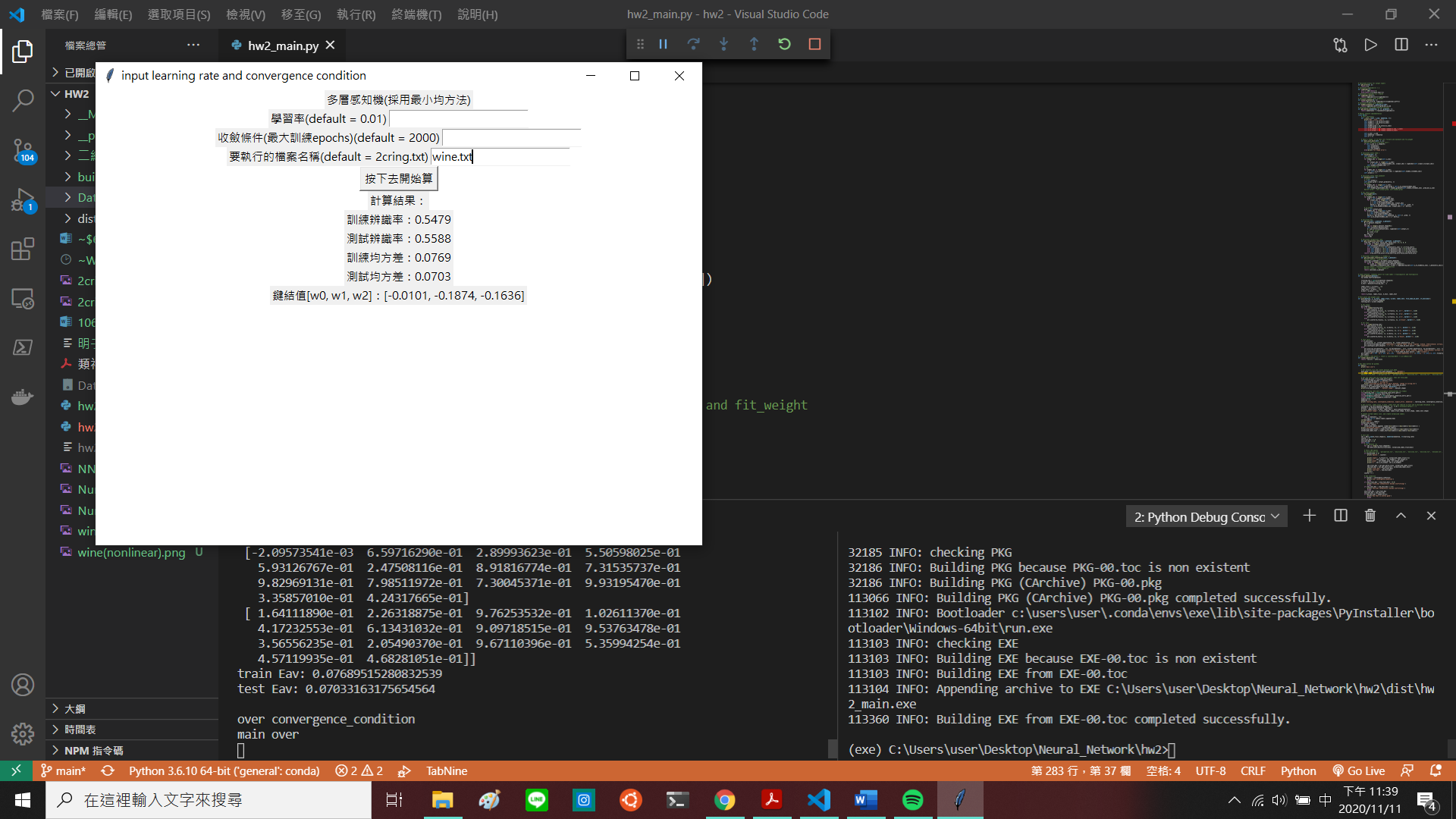
1. 如有加分項目，請在報告中說明

本感知機應可推展至多維的x資料集(但是無法視覺化呈現)，實驗結果如下：

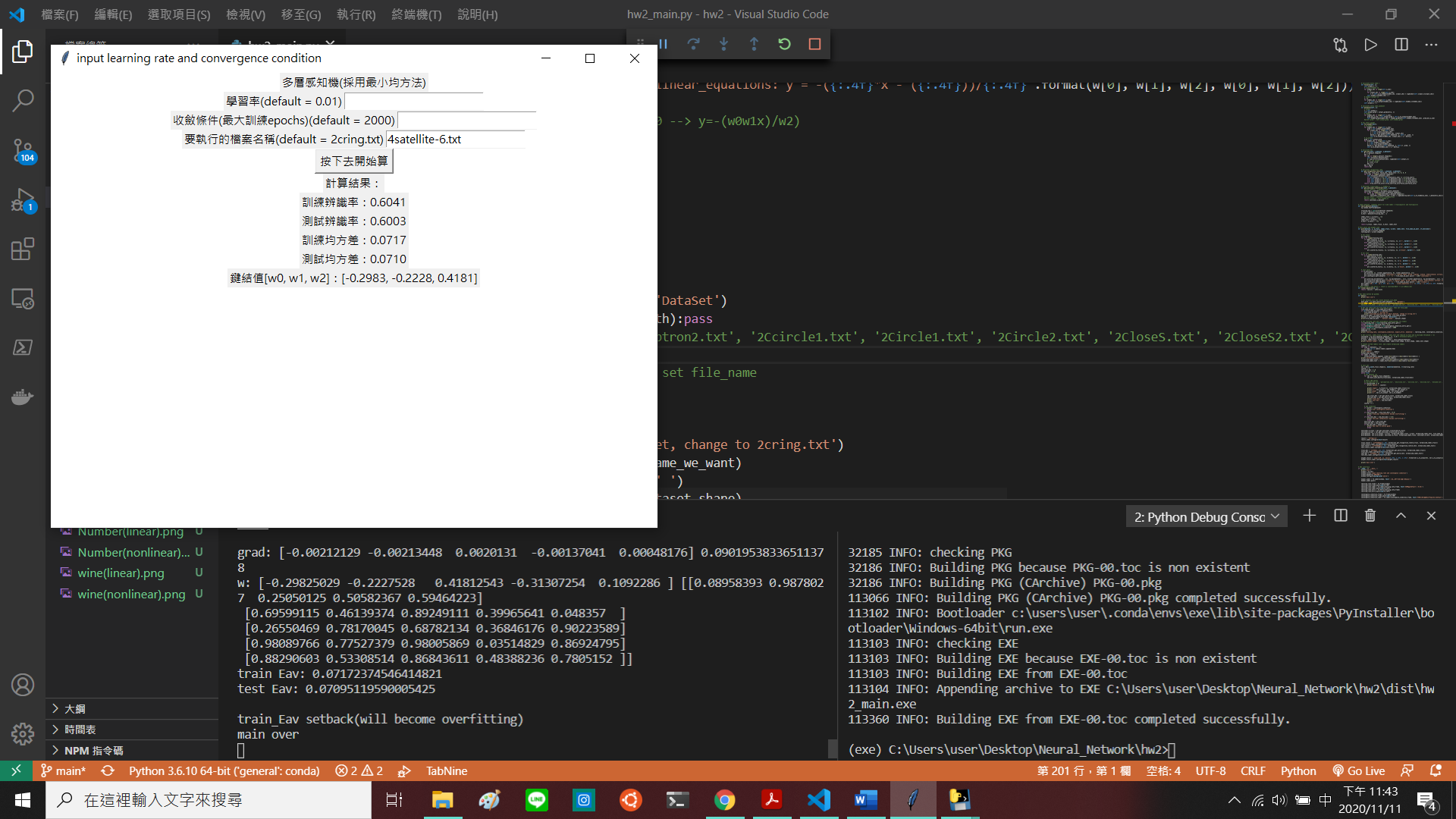
Number.txt(train:0 test:1)



wine.txt(train:0.54 test:0.55)



4satellite-6.txt(train:0.6 test:0.6)



5CloseS1.txt(train:0.975 test:0.983)

