機器學習 HW3 – 程式報告　111652015 曹晉嘉

(一) 目的 ：

甲、以上次的程式碼架構，多預測目標函數的微分值

(二) 方法 ：

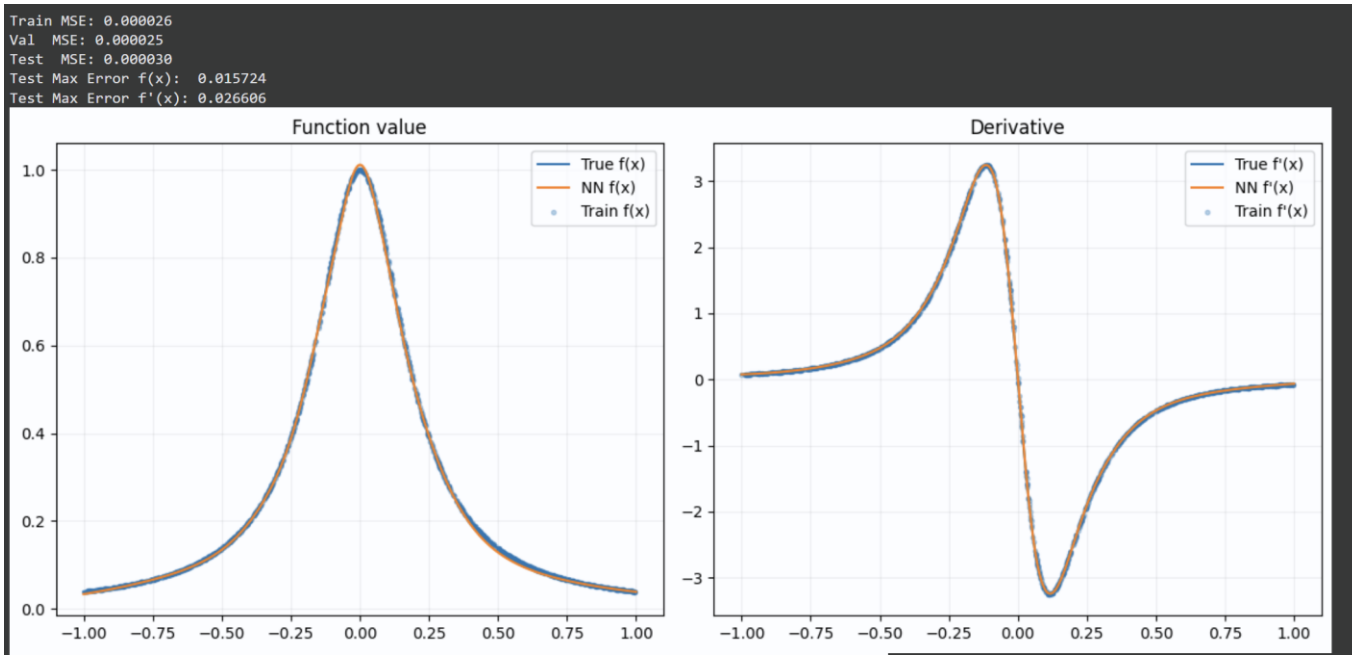
甲、使用四層神經網路 (中間的兩層隱藏層依舊都使用 10 個神經元)

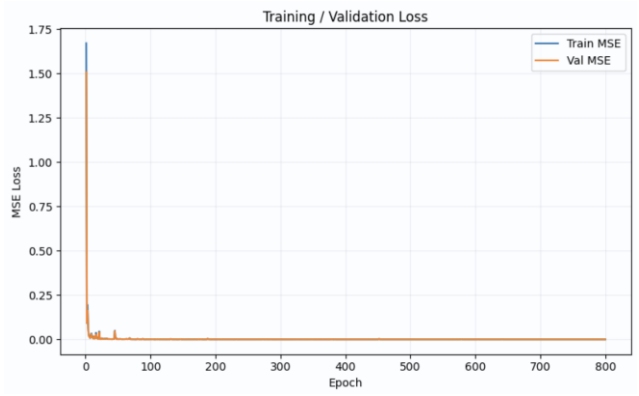
乙、輸出從原本的一個數字 (函數預測值)，改為一個向量 (函數預測值 ，微分後函數預測值)

丙、超參數 (總資料數 = 2000，學習率 = 0.1，訓練輪數 = 800，每批次大小 = 1) ，其中訓練資料占總資料數六成、驗證資料占三成、測試資料占一成

丁、啟動函數依舊是 sigmoid(x) = 1 / (1+e^-x)

(三) 結果 ：





(四) 討論 ：

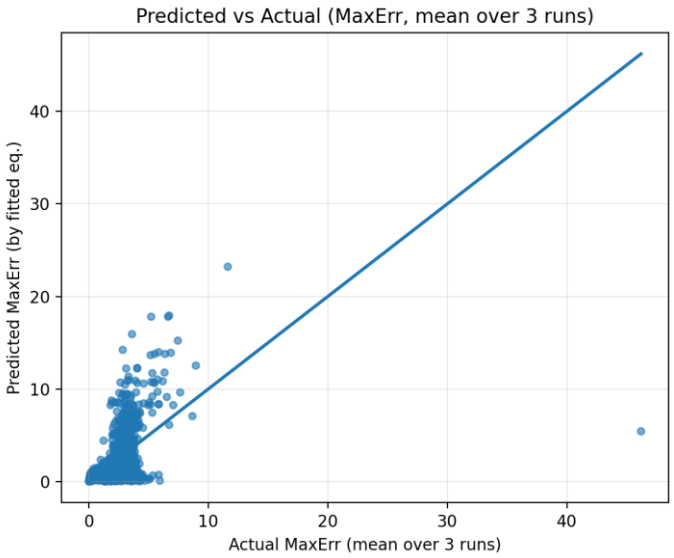
甲、為了了解用什麼超參數可以訓練出誤差最小的神經網路，因此我將四個超參數分別取下列的值(如下圖)，做成網格讓電腦使用下列所有超參數，並給出訓練出的神經網路與目標函數的最大誤差

```
SEARCH_SPACE = {
    "N": [100, 200, 300, 500, 1000, 1500, 2000],      #總資料
    "lr": [0.001, 0.005, 0.01, 0.02, 0.05, 0.08, 0.1, 0.3],      #學習率
    "epochs": [10, 30, 50, 100, 200, 400, 600, 800, 1000],      #訓練輪數
    "batch_size": [1, 4, 8, 16, 32, 64, 128],      #每批次大小
}
```

在執行一輪後，我發現在許多訓練組數較少的情況下，即使使用一樣的超參數，每次訓練後得到的誤差也時好時壞，故我讓電腦在使用每個超參數都訓練出三個神經網路，並將此超參數的誤差定義為三個神經網路與目標函數的最大誤差的平均值，並將最終得到的能使誤差最小的超參數帶入並匯出在上方（ 第三節結果 ）那邊所顯示的那兩張圖。

乙、我試圖用 python 裡面的分析工具匯出以四個超參數為變數並輸出訓練的神經網路與目標函數的誤差，得以下式子但效果並不好(如下圖所示)

$$\text{MaxErr} \approx 20.96 \times N^{-0.3817} \times \text{lr}^{-0.1636} \times \text{epochs}^{-0.4597} \times \text{batch\_size}^{0.3687}$$



下次可以嘗試取得更多資料點或使用其他工具來擬合資料，或許可以得到更精確的式子。