網路shape:

用三個距離當作input，第一層12個神經元，第二層6個神經元，輸出層一個神經元，使用sigmoid函數，輸出為(0~1)，再把輸出投影到-40~40之間。

另外，因為當初作業一在做的時候就是mlp了，只有把relu改成sigmoid，因為我希望輸出在0~1之間，這樣比較好處理，要看詳細的話可以看我作業一的檔案



訓練資料處理:

有嘗試過的方法如下

1.不處理直接丟進去 -> 撞牆

2.用min跟max 線性投影到0-1之間 -> 撞牆

3. fuzzy.zmf 參數測試了好幾個 -> 差點成功，撞牆

4. 正規化 -> 撞牆

最後用的方法是，左、右45度、前方距離分開討論，先計算出標準差、平均數，左右45度大於平均值的直接為0、前方距離大於平均值的三分之二為0，想法是因為距離都很遠都不用考慮撞牆，輸入就不需要有差異，其餘就是短時間內有可能會撞的情況，對資料標準化後加上負號(即越靠近牆值越大)。

標準化後，把三個list中的所有資料除上最大元素 ，即可把所有資料轉換到0~1之間

碰撞、碰到終點線:

碰撞偵測、抵達終點線我是用sympy這個python庫做的，車是一個圓形，牆壁是一個線段，每前進一格用for迴圈跑一遍看有沒有牆跟車子超過兩個交點就可以了

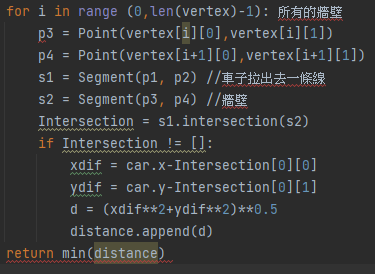
距離偵測:

用車子的距離跟方向，即可得出一個距離很遠的點(至於多遠可以調整，只要讓程式可以正確運行即可)，兩個點可得出一個線段



判斷此線段跟所有牆壁有無交點，交點可以進一步求出距離，其中最小值即為所求

一樣是直接拿sympy來用



Gui操作:

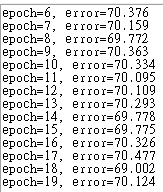
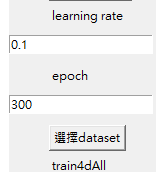


點選建立新網路，會產生一個類神經網路，儲存在叫model的檔案

設定lr、epoch、選擇trainset後

如果目錄下已有model，可以不用建立新的網路

按開始訓練會讀取model，開始訓練類神經網路



如果訓練的結果不佳，更改lr、epoch即可繼續訓練，不需要建立新的網路(當然也可以)

訓練完後會產生檔案儲存中位數、標準差、最大值

因為是用讀取檔案的方式，只要目錄下訓練好的model，那以上步驟都可以選擇性跳過，直接選擇軌道檔案開始預測



選擇要讀取的檔案後，會畫出軌道and終點線，產生一台車，計算出初始的三個距離，按下START會畫出車的移動軌跡，抵達終點線後會產生要求的軌跡檔案

因為這個gui套件預設只有單線程，如果發生意外停止回應可以等他一下