*Code 由 Chatgpt 和 Gemini 生成

(1) 資料轉換

將下載的資料依據描述加上經緯度資料，做成兩筆訓練用資料，分別為 X_class, y_class (分類資料集，X 為經緯度，y 為 label)和 X_reg, y_reg (回歸資料集，X 為經緯度，y 為 value)。

再將兩筆資料各自分為訓練集和測試集 (train : test = 4 : 1)，Xc_train, Xc_test, yc_train, yc_test 為分類資料集，Xr_train, Xr_test, yr_train, yr_test 為回歸資料集。

(2) 模型訓練

起初的架構就是直接把資料餵給神經網路，後來發現這樣的效果不佳，所以再模型的最前面加了一層 normalizer，對 X 的資料 normalize (使用 Xc_train\Xr_train 的平均值和標準差，確保沒有 test 的資料外洩)。

接著兩個模型都會經過三個全連接層，神經元個數依序是 64、32、16，Activation function 皆為 ReLU。

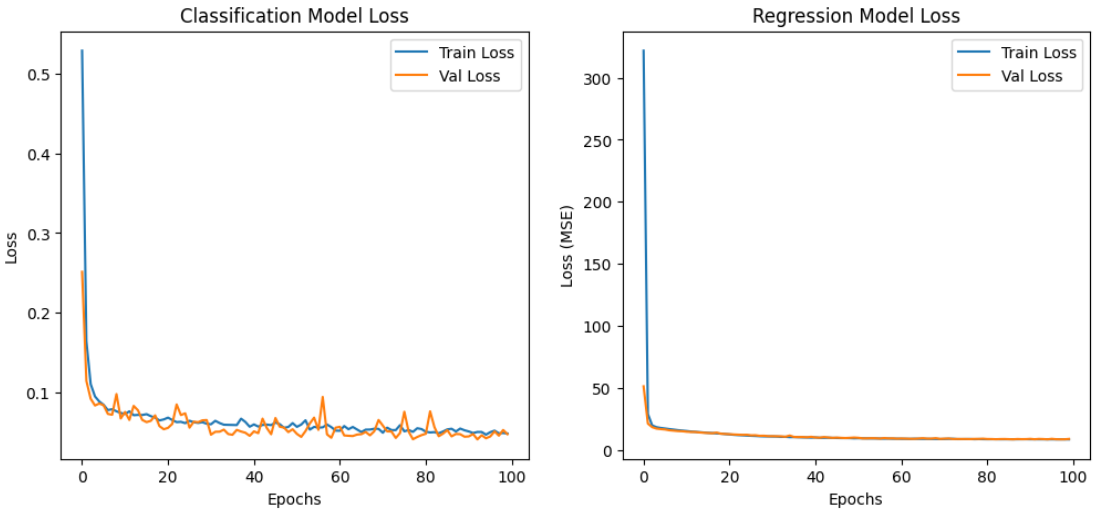
分類模型最後輸出前會經過 sigmoid function 並輸出 0 到 1 之間的值，代表模型對該點預測出的機率，<0.5 代表模型預測是 0，>0.5 則是 1。

分類模型的其他參數: optimizer=Adam, loss='binary_crossentropy', learning_rate=0.001, epochs=100, batch_size=32

回歸模型在輸出層不會經過 activation function，直接輸出模型對該點的預測值。

回歸模型的其他參數: optimizer=Adam, loss='mse', learning_rate=0.001, epochs=100, batch_size=32

(3) 結果



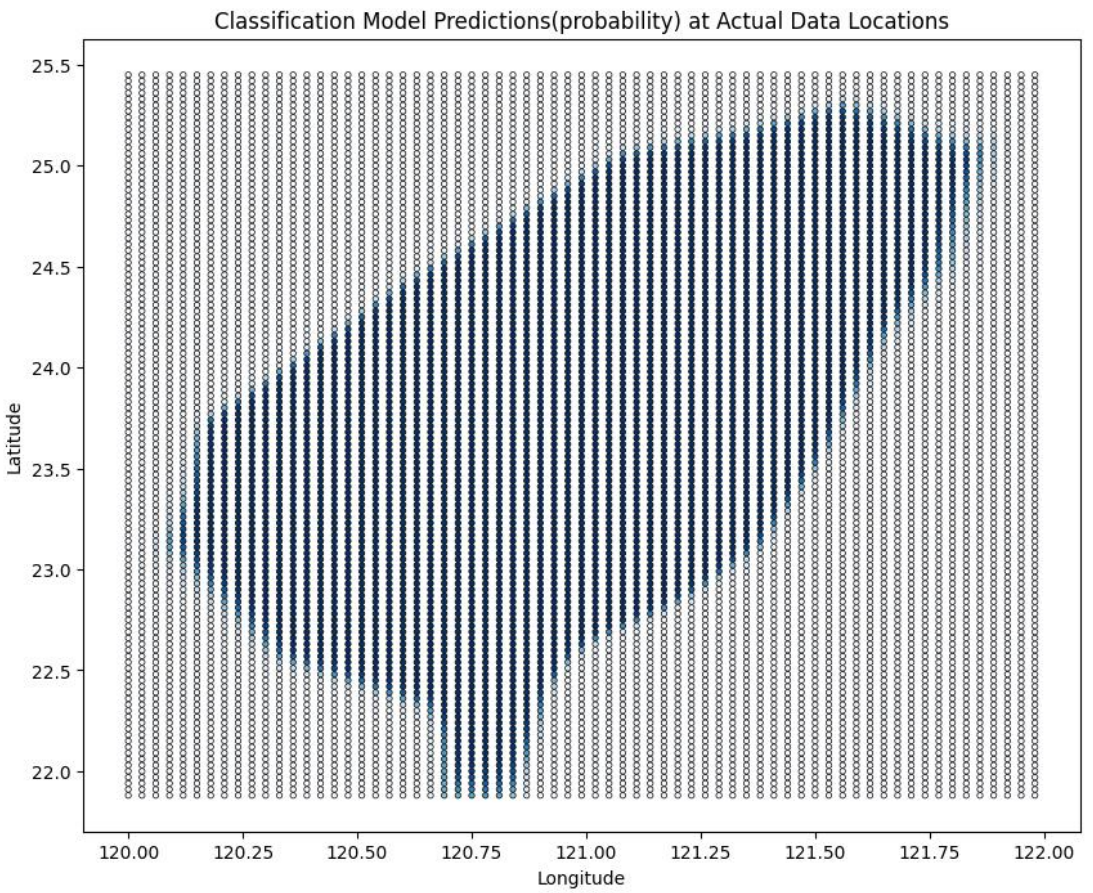
兩個模型都在一開始就以非常快的速度收斂，後面 loss 下降的速度非常緩慢，所以推測再增加 epoch 表現也不會提升多少了。

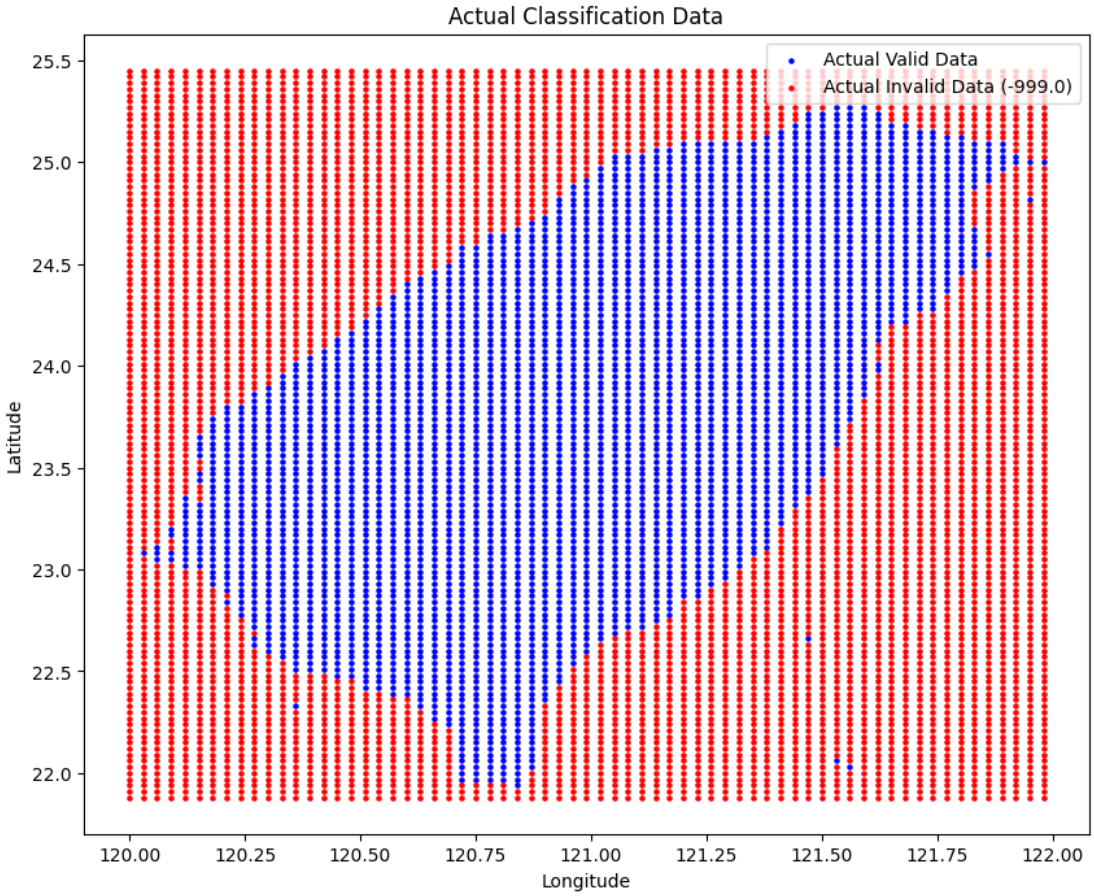
Classification Model ： Test Loss 0.0478，預測正確率 97.89%

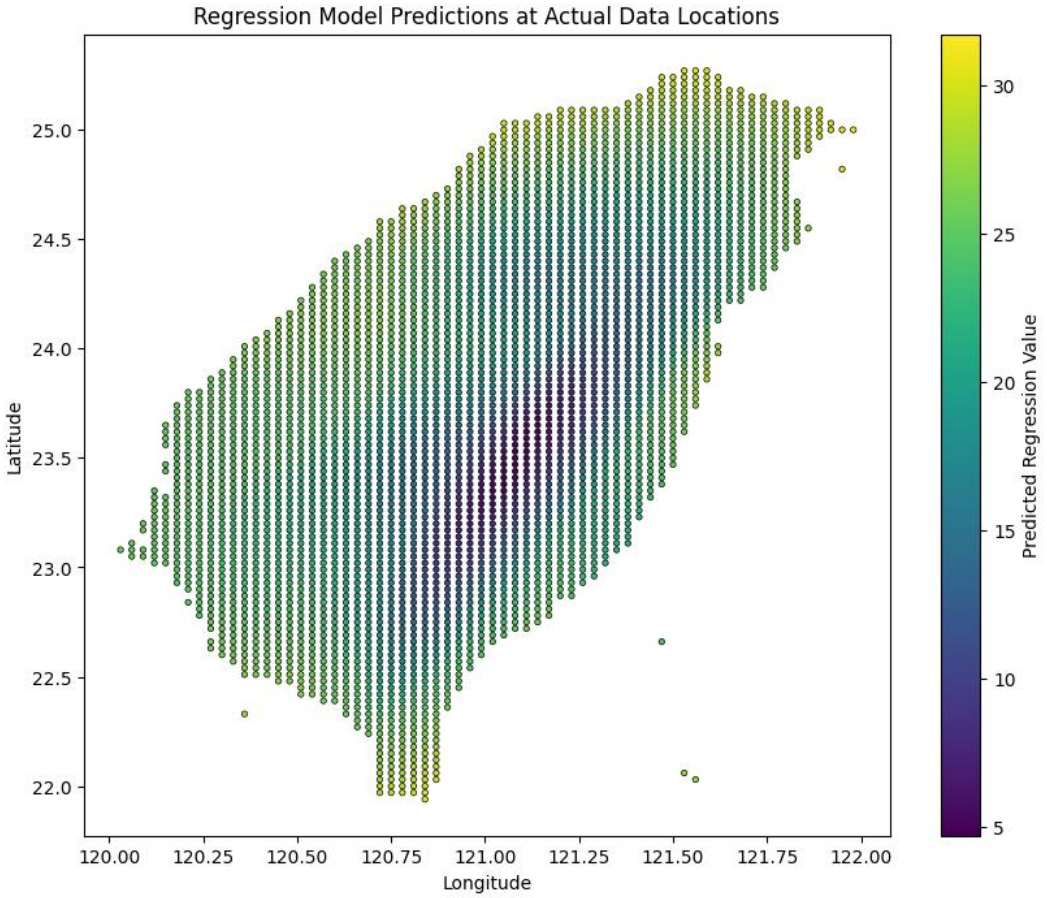
Regression Model ： Test Loss (MSE) 8.8491，平均溫度絕對誤差 2.1470 度

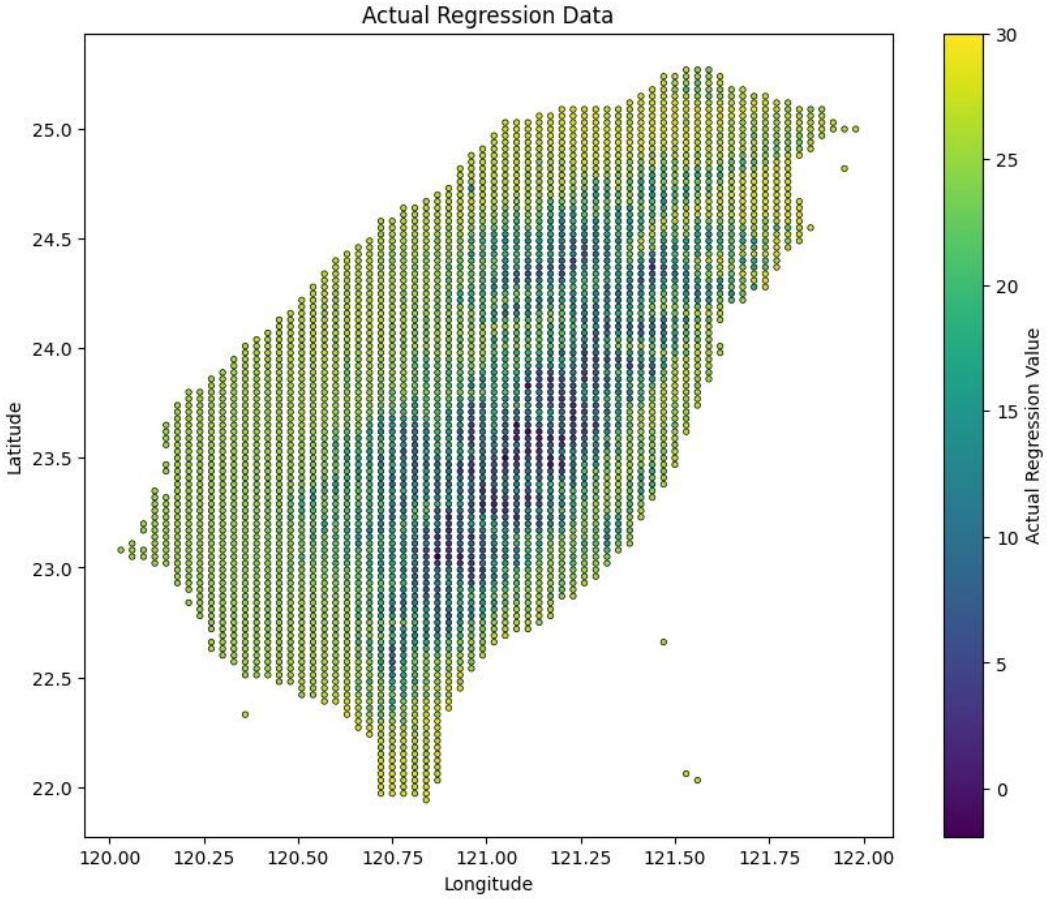
從最後在測試集上的表現來看，分類模型和回歸模型都表現得不算差，不過也都還有進步空間，尤其是回歸模型，2 度的誤差放到現實來講還是蠻有感的。

(預測結果與真實資料比對圖在下一頁)


Classification Model Predictions(probability) at Actual Data Locations
Latitude
Longitude
25.5
25.0
24.5
24.0
23.5
23.0
22.5
22.0
120.00
120.25
120.50
120.75
121.00
121.25
121.50
121.75
122.00


Actual Classification Data
Actual Valid Data
Actual Invalid Data (-999.0)
Latitude
Longitude
25.5
25.0
24.5
24.0
23.5
23.0
22.5
22.0
120.00
120.25
120.50
120.75
121.00
121.25
121.50
121.75
122.00


Regression Model Predictions at Actual Data Locations
Latitude
Longitude
Predicted Regression Value
25.0
24.5
24.0
23.5
23.0
22.5
22.0
120.00
120.25
120.50
120.75
121.00
121.25
121.50
121.75
122.00
30
25
20
15
10
5


Actual Regression Data
Latitude
Longitude
Actual Regression Value
25.0
24.5
24.0
23.5
23.0
22.5
22.0
120.00
120.25
120.50
120.75
121.00
121.25
121.50
121.75
122.00
30
25
20
15
10
5
0