Machine Learning
Assignment 6

CHANG Yung-Hsuan (張永璿)
111652004
eiken.sc11@nycu.edu.tw

October 10, 2025

1. There are unanswered questions from the lecture, and there are likely more questions we haven't covered. Take a moment to think about these questions. Write down the ones you find important, confusing, or interesting.

   Answer. I am curious about the role of regularization in neural networks. In linear models, ridge regression is commonly used to mitigate overfitting by penalizing large coefficients. I wonder whether similar techniques exist for neural networks—methods that modify the loss function to constrain or regulate the magnitude of weights within a reasonable range. Or, are such models primarily guided by empirical tuning of architectures and parameters rather than explicit analytical regularization?

2. (Classification using GDA) Your task is to use Gaussian Discriminant Analysis (GDA) to build a classification model. To complete this assignment, make sure you:

   a. Write your own code to implement the GDA algorithm.

   b. Clearly explain how the GDA model works and why it can be used for classification, in particular this data set.

   c. Train your model on the given dataset and report its accuracy. Be explicit about how you measure performance (e.g., accuracy on a test set, cross-validation, etc.).

   d. Plot the decision boundary of your model and include the visualization in your report.

   Solution.

a. See `temperature.py` in `Week_6`.

b. The Gaussian Discriminant Analysis model assumes that the conditional distribution of the features given the class label is multivariate normal, i.e.,

$$p(x \mid y = k) \sim \mathcal{N}(\mu_k, \Sigma_k),$$

where μ_k and Σ_k are the class-specific mean vector and covariance matrix.

Using Bayes' rule, the posterior probability is

$$p(y = 1 \mid x) = \frac{p(x \mid y = 1) \cdot p(y = 1)}{p(x \mid y = 1) \cdot p(y = 1) + p(x \mid y = 0) \cdot p(y = 0)}.$$

A new data point x is classified as valid when this posterior probability exceeds a threshold τ.

I think GDA might not be suitable here because the boundary between valid and invalid regions in longitude–latitude space is not smooth and not elliptical at all. I guess the instructor just wants us to practice coding from scratch and to be familiar with GDA.

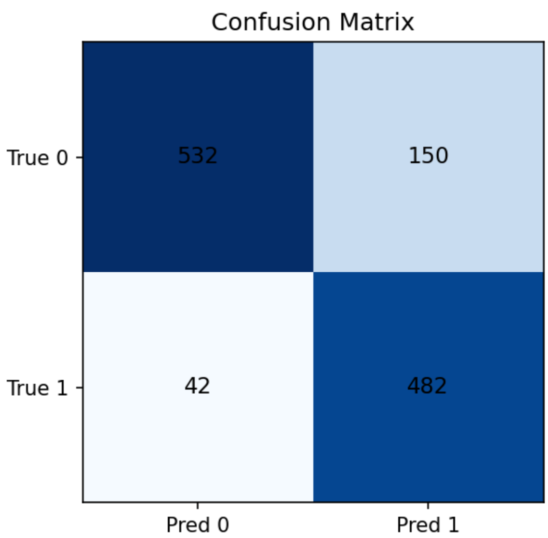

Figure 1. Confusion Matrix for the QDA Model

c. To be concise, I split the dataset with a stratified manner, training 70%, testing 15%, and validation 15%. Threshold τ is tuned on the validation set to maximize **accuracy**, and final performance is reported on the test set. Eventually, I have the threshold 0.4 with validation accuracy 85.32% and testing accuracy 84.08%. See Figure 1 for the confusion matrix.

d. The model achieves an accuracy of 84.08% on the test set; however, the high accuracy meant nothing here. A higher false positive rate occurs near the north-east and south-west part; just like I said, the Taiwan island is not elliptical so QDA is not suitable. See Figure 2 for the visualization.

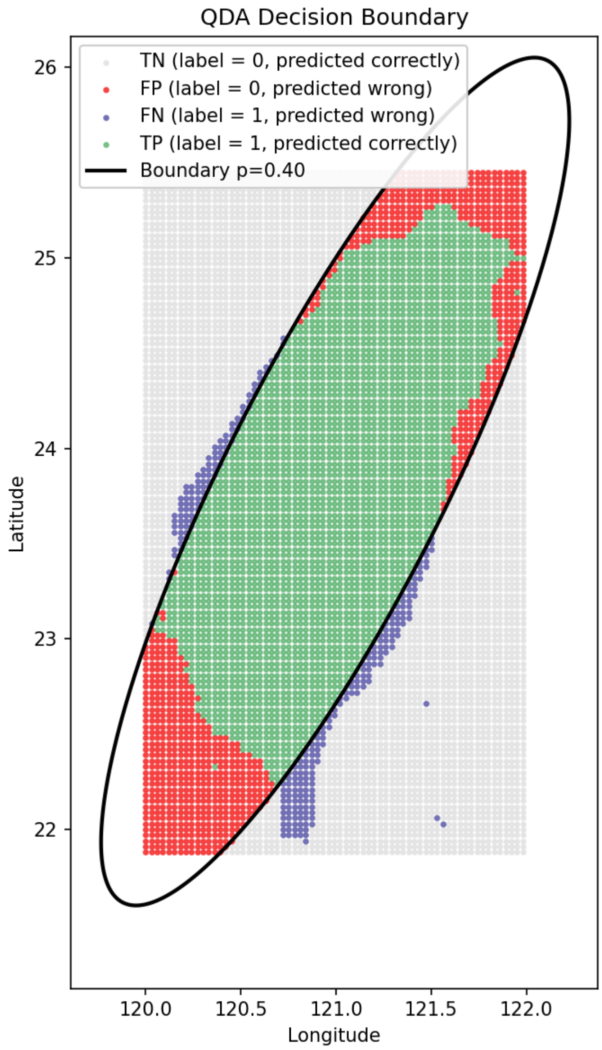

Figure 2. The Decision Boundary for the QDA Model with Threshold 0.40

3. (Regression) Your task is to build a regression model that represents a piecewise smooth function. To do this, combine the two models from Assignment 4 into a single function. Specifically, let

- $C(\vec{x})$ be your classification model, and
- $R(\vec{x})$ be your regression model. Then construct a model $h(\vec{x})$ defined as

$$h(\vec{x}) = \begin{cases} R(\vec{x}), & \text{if } C(\vec{x}) = 1; \\ -999, & \text{if } C(\vec{x}) = 0. \end{cases}$$

To complete this assignment, make sure you:

a. Implement this combined model in code.

b. Apply your model to the dataset and verify that the piecewise definition works as expected.

c. Briefly explain how you built the combined function.

d. Include plots or tables that demonstrate the behavior of your model.

Solution.

a. See `enhanced_temperature.py` in `Week_6`. I construct the combined function as follows:

   - $C(\vec{x})$: a decision tree model with `max_depth=40` on `(longitude, latitude)` with a stratified split, training 70%, testing 15%, and validation 15%. I tune the probability threshold τ on the validation set to maximize **accuracy**, then freeze it for testing and for the piecewise mask. See Figure 5 for the decision boundary.
   - $R(\vec{x})$: a two-layer neural network with 64 neurons each layer and ReLU activation functions trained on valid points only, with standardized inputs, Adam optimizer, and early stopping. See Figure 6 and Figure 4 for the regression.

     The neural network actually did a really good job to be a classifier and a regressor at the same time as Figure 6 showed that around the seashore are all blue. I just used a big cannon to shot down a bird.
   - $h(\vec{x})$: evaluate $p = P(C = 1 \mid x)$, set `valid-mask` $= \mathbb{I}_{p \geq \tau}$; return `R(x)` on the mask and `-999` elsewhere. See Figure 7 for the result and Figure 8 for the error.

     For the error, I was trying to plot only test set, but it turns out that Figure 4 suggests the regression is quite bad (the dots are not collapsing to a single curve), so I just plotted all of them anyway.

b. The piecewise heatmap shows valid temperature only within the tuned decision boundary; all other cells are `-999` (masked).

c. I first fit the decision tree model on all points `(lon, lat)` with labels `{0,1}`; tune τ on validation. Then, I fit the neural network on valid points only to get $R(\vec{x})$. Then, for any input $\vec{x}$, if $P(C = 1 \mid \vec{x}) \geq \tau$, output $R(x)$; else output `-999`. So two models are trained separately then combined.

d. The test confusion matrix indicates an **accuracy** of about 98.18%, meaning the geographic mask closely follows the valid region. See Figure 3 for the confusion matrix.

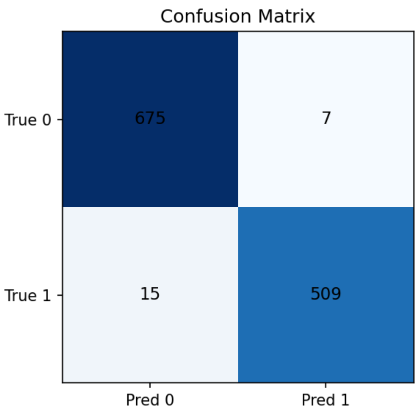

Figure 3. Confusion Matrix of the Decision Tree Model

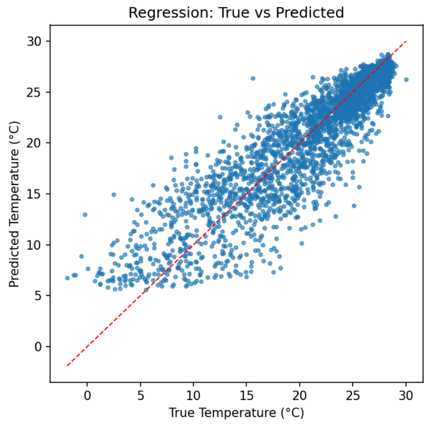

Figure 4. Neural Network Regression: True vs. Predicted (invalid omitted)

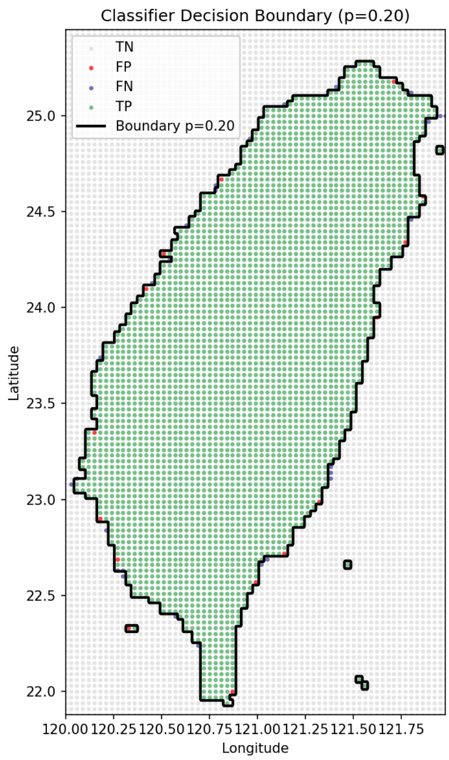

Figure 5. Decision Boundary at $p = 0.20$

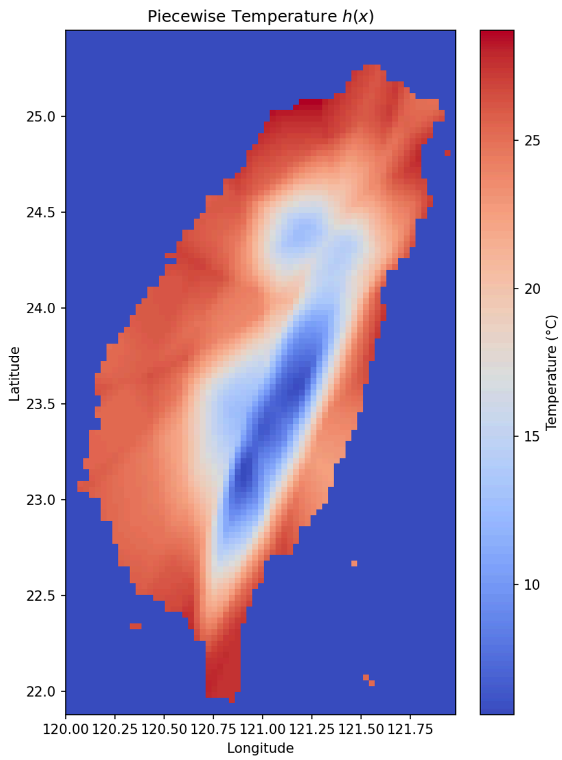

Figure 6. Neural Network Regression

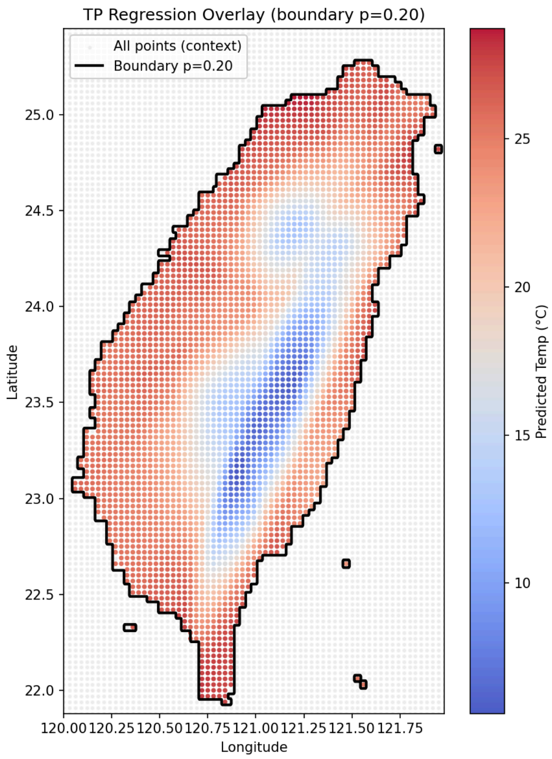

Figure 7. TP Regression Overlay (colors = predicted temperature; boundary $p = 0.20$)

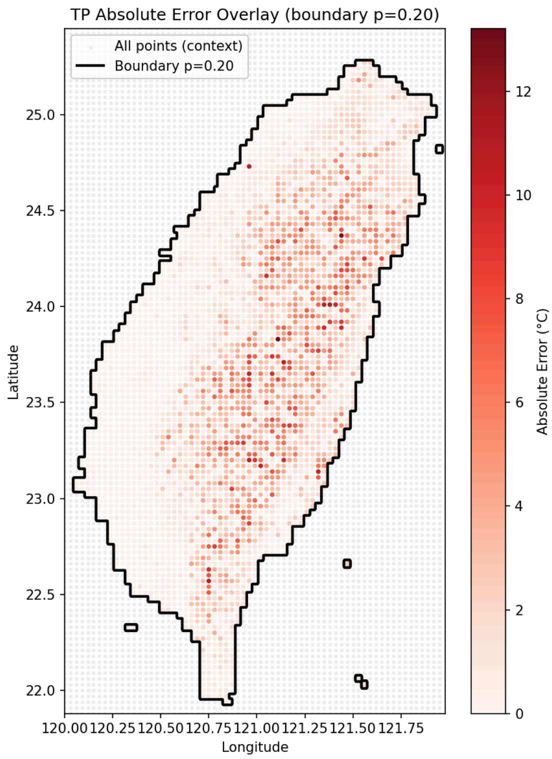

Figure 8. TP Absolute Error Overlay (white = 0, red = larger error; boundary $p = 0.20$)