Homework week 6

Programming 1

Classify grid cells as valid vs invalid using two features: Longitude and Latitude(see Figure 1).

Implementation I coded GDA in NumPy: estimate class priors, class means, and 22 covariance matrices from the training split; score new points with the class likelihoods; predict valid when the class-1 probability greater than 0.5.

Boundary plot I evaluate the fitted GDA on a dense lon–lat grid and draw the 0.5 contour in red over the data (see Figure 2). Most valid points are enclosed, with a few exclusions near complex borders. Notes that the small neural model (2 inputs, two 16-unit hidden layers, 1 output) better hugs irregular boundaries, which explains any accuracy gap vs. GDA. For further detail see https://colab.research.google.com/drive/1CxdY2q_6oZUZtIBzCWXiMGhwatjSYwOq#scrollTo=6gy8GMIsridC.

Programming 2

Objective Build a regression model that is smooth where observations are valid and constant elsewhere. Here, we define $C(x)$ be a classifier that predicts whether a location has a valid temperature and $R(x)$ be a regressor that predicts temperature on valid locations, and

$$h(x) = \begin{cases} R(x), & \text{if} \quad C(x) = 1, \\ -999, & \text{if} \quad C(x) = 0. \end{cases} \tag{0.1}$$

I model the temperature field as a piecewise-smooth function by combining a mask classifier with a temperature regressor. Each grid cell is described by longitude and latitude; cells with observed value equal to -999 are treated as valid, others as invalid. Let $C(x)$ denote the probability that a location is valid and $R(x)$ the temperature predicted on valid locations. I define $h(x)$ in words as: return the regressor's output inside the classified coverage region and a constant -10 outside (-10 just for convenience). This keeps the surface smooth where we have data support and flat elsewhere, with a single controlled jump along the boundary.

Both components are implemented from scratch in NumPy. The classifier $C(x)$ is a small MLP with two inputs, two hidden layers of 16 neural networks ($2- > 16- > 16- > 1$). Optimization is Adam with binary cross-entropy for C and mean squared error for R.

In conclusion, the construction separates where to predict from what to predict: C learns the geographic support and R learns the temperature field within it. The outside value of -10 keeps the dynamic range moderate compared with using -999, which improves readability in maps and slices (see figure 5 when

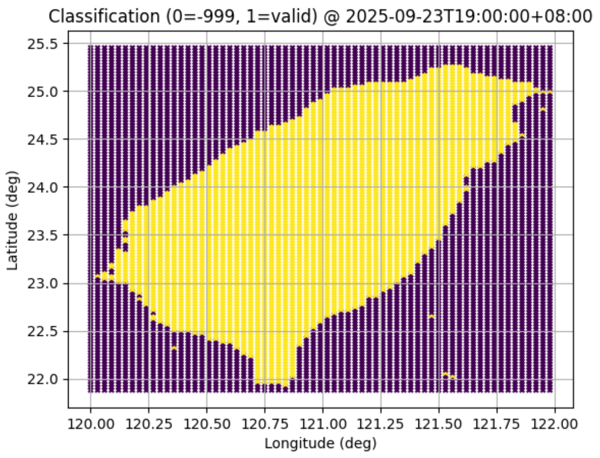

Figure 1:

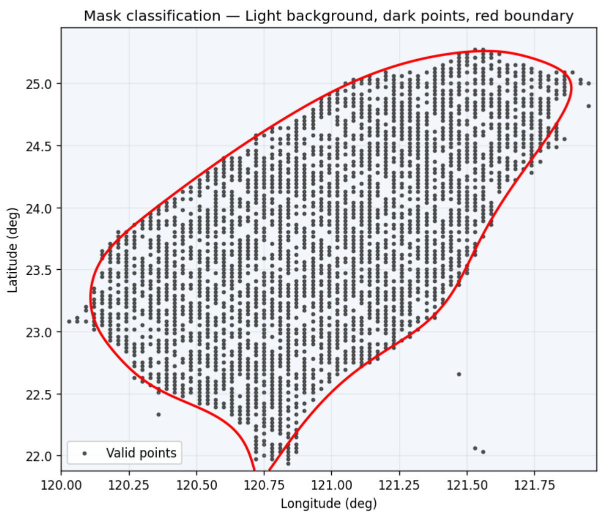

Figure 2:

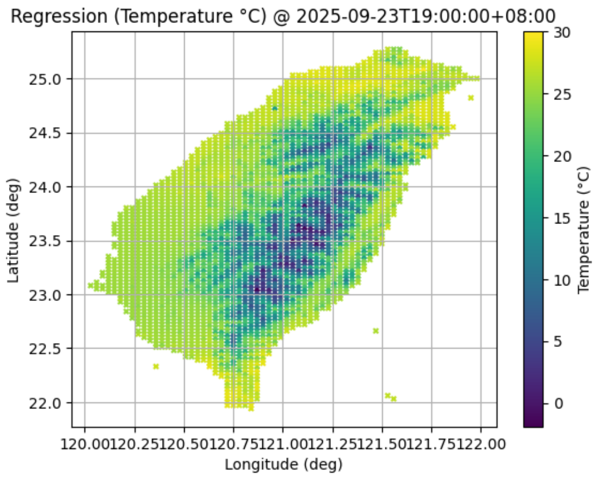

Figure 3:

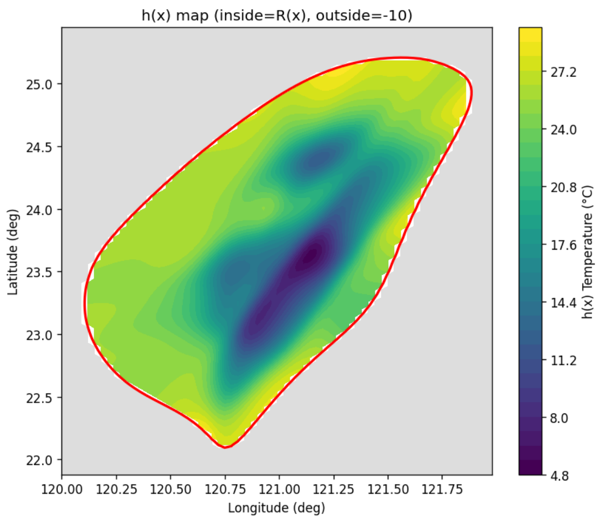

Figure 4:

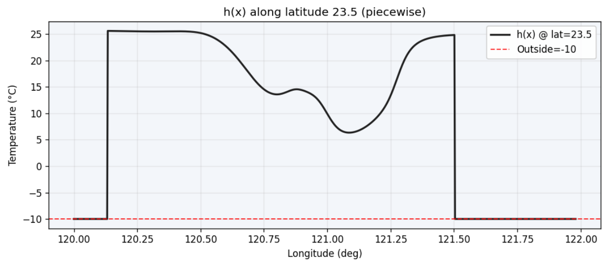

Figure 5: latitude at 23.5 with x=longitude y=value of h

latitude equal to 23.5). If needed, the tightness of the coverage and the visual sharpness of the jump can be tuned by adjusting the threshold, hidden size, training epochs, or the outside constant. For further detail see https://colab.research.google.com/drive/1G2mcaOxnqgU_rxiA-nehcftOhEbYgJKf#scrollTo=j2SSGw2U895M.

Problem 3

Question