Weather Grid Classification and Regression Using Neural Networks

Chia-Yu Ou

September 30, 2025

1 Introduction

This report presents a two-part supervised learning task based on gridded hourly temperature observation data from a meteorological platform. The dataset is structured as a 67 (longitude) × 120 (latitude) matrix where each grid point contains a temperature value in °C. The resolution is 0.03 degrees in both longitude and latitude. Invalid values are marked as -999. We perform:

- Classification: Determine whether a grid value is valid.
- Regression: Predict the actual temperature value at each valid location.

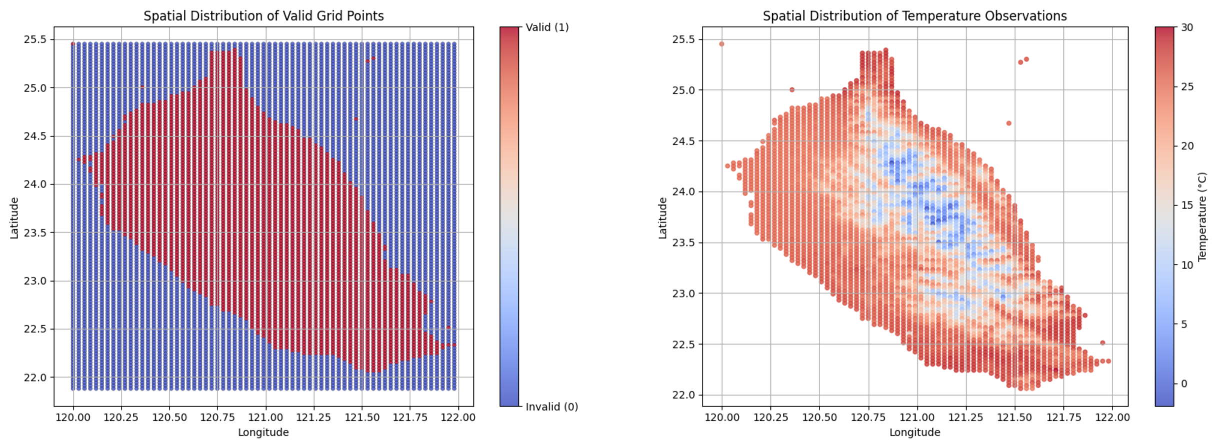
2 Dataset

2.1 Data Conversion

Using `preprocess.py`, we transformed the raw temperature matrix into two datasets:

1. **Classification dataset**: $(\text{lon}, \text{lat}, \text{label})$, where label $= 1$ if temperature $\neq -999$, else 0.
2. **Regression dataset**: $(\text{lon}, \text{lat}, \text{value})$, where only valid temperature values are retained.

We use NumPy and pandas to transform the gridded temperature data into a flattened DataFrame format, preserving floating-point precision during the conversion. The resulting dataset includes longitude, latitude, and observed temperature values. To gain spatial insights, we visualize the regression data using a scatter plot where each point represents a grid location colored by its corresponding temperature.



(a) Spatial Distribution of Valid Grid Points (b) Spatial Distribution of Temperature Observations

Figure 1: Visualization of grid point validity and corresponding temperature observations across Taiwan. Red indicates valid or higher temperature areas, and blue indicates invalid or cooler regions.

2.2 Feature Standardization

Prior to training, we apply feature standardization to the input features (longitude and latitude) using `StandardScaler` from `scikit-learn`. This transformation ensures that both features have zero mean and unit variance, which helps accelerate training convergence and improves model stability. The same scaler is applied to the validation and test sets to maintain consistency.

2.3 Data Splitting Strategy

This work employs a 60%/20%/20% split for training, validation, and testing. The classification dataset is divided using stratified sampling to maintain the proportion of valid and invalid labels across all subsets. This stratification is crucial given the spatial imbalance in the data.

3 Classification Model

3.1 Model Assumptions

We define a binary classifier using a feed-forward neural network with the following structure:

- Input layer: 2 features (longitude and latitude)
- Hidden layers: [64, 32, 16] with ReLU activations
- Output layer: 1 node with Sigmoid activation

The model predicts the probability that a grid point is valid. We minimize binary cross-entropy loss:

$$\mathcal{L}(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

Optimization is done via Adam with learning rate 10^{-3}.

3.2 Training and Evaluation

The dataset was split into training (60%), validation (20%), and testing (20%) sets. Input features were standardized using `StandardScaler`. The training process exhibited steady convergence, as evidenced by the decreasing training and validation losses over epochs.

```
Epoch 0   | Train Loss: 0.6854 | Valid Loss: 0.6823
Epoch 100 | Train Loss: 0.2121 | Valid Loss: 0.1996
Epoch 900 | Train Loss: 0.0396 | Valid Loss: 0.0414
```

Final test accuracy: **0.9807**. Despite the relatively simple inputs, the network is able to roughly approximate the shape of Taiwan (valid vs. sea).

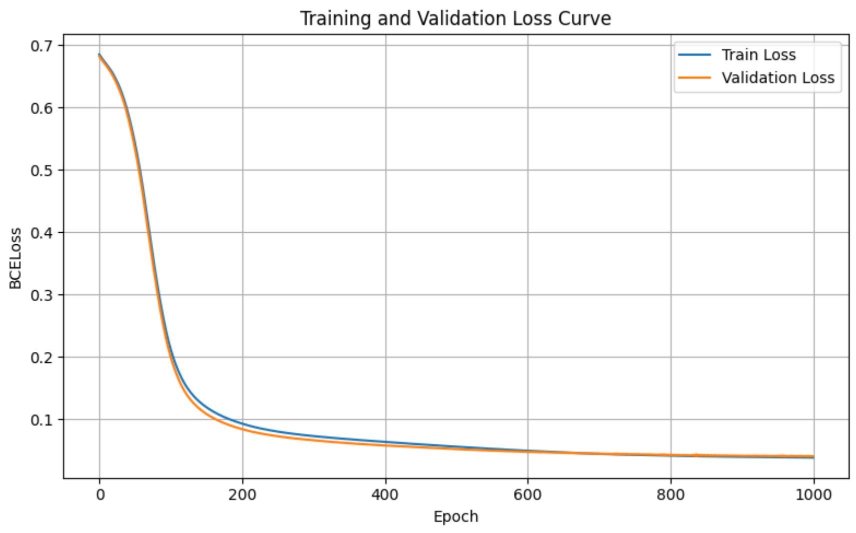


Figure 2: Training and validation loss over epochs. The model shows steady convergence with no signs of overfitting.

4 Regression Model

4.1 Model Assumptions

We define a regression model using a feed-forward neural network with the following architecture:

- Input layer: 2 features (longitude and latitude)
- Hidden layers: [128, 64, 32, 16] with ReLU activations
- Output layer: 1 node with linear activation

The model predicts the temperature value at a given location. We minimize the Mean Squared Error (MSE) loss:

$$\mathcal{L}(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Optimization is performed using the Adam optimizer with a learning rate of 10^{-2}. The model is trained for 1000 epochs with early monitoring on the validation loss.

4.2 Training Result

```
Epoch 0   | Train Loss: 504.1043 | Valid Loss: 483.4955
Epoch 100 | Train Loss: 14.6084    | Valid Loss: 17.1549
Epoch 900 | Train Loss: 7.3502    | Valid Loss: 9.5827
```

Final test MSE: **6.473**. The model still captures the overall spatial distribution of temperature and demonstrates the potential of neural networks in learning spatial patterns.

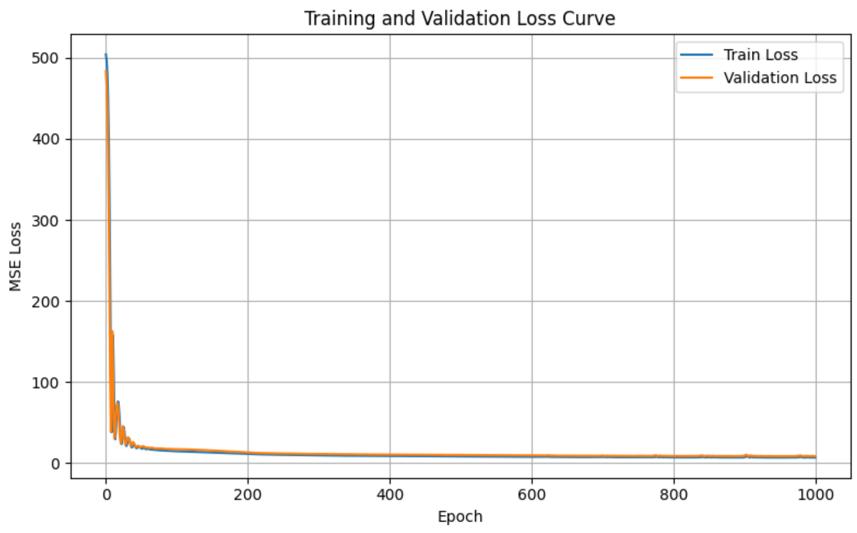


Figure 3: Training and validation loss over epochs. The model shows steady convergence with no signs of overfitting.

5 Conclusion

Both models demonstrate strong performance using only basic geographic features. The classification model achieves a high test accuracy of 0.9807 and successfully distinguishes between valid and invalid grid points, roughly approximating the outline of Taiwan's landmass. The regression model achieves a final test mean squared error (MSE) of 6.473, indicating reasonable predictive capability for surface temperature estimation based on location alone.