Assignment4

September 2025

1 Question

I found that the choice of the training set, test set and validation set will affect the result. Since the form of the data in the xml file, I can't just use the first 80 percent of the data to do the choice to get a good result, so I choose randomly. I'm wondering that if there is any other way to choose these data set.

2 Programming report

First, we change the xml file to two sets of data that denoted $df_{regression}$ and $df_{classification}$ which are in the form of (lon,lat,temperature (value)) and (lon,lat,0(if temp = -999) or 1 (label)) respectively.
And we want to train two models which predict the temperature and classify if the label are 0 or 1 respectively.

Then I choose 80 percent data of $df_{regression}$ randomly as training sets x_{train} and y_{train} and the other 20 percent of data as test sets x_{test} and y_{test}.(I first try choosing first 80 percent of data in the set and the result is bad. I think that it's because of the other 20 percent of data are the same value. So I try choosing randomly.)

Then for the validation sets x_{val} and y_{val}, I choose 20 percent of data from x_{train} and y_{train}.

Then I scale x_{train}, x_{test} and x_{val}.

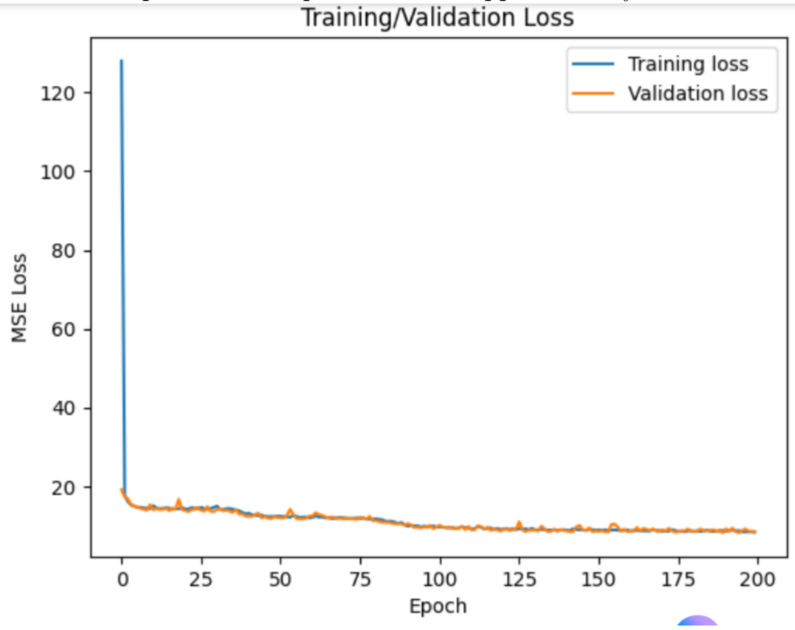
Then I design a neural network with 2-neuron which takes in $x = (lon, lat)$ in the input layer, 32-neuron with the relu activation function in the first and second hidden layer, then the output layer has 1-neuron outputs the temperature as the hypothesis function.

I choose MSE as the loss function.

Then to train the neural network, it will do a forward and back propagation for

200 epochs while using the mini-batch gradient descent for 32 data each time and Adaptive moment estimation as optimizer with learning rate 0.001.

Next, we have the following result: MSE error: 8.619676226402104 which implies that the predicted temperature have approximately 2.93 error.



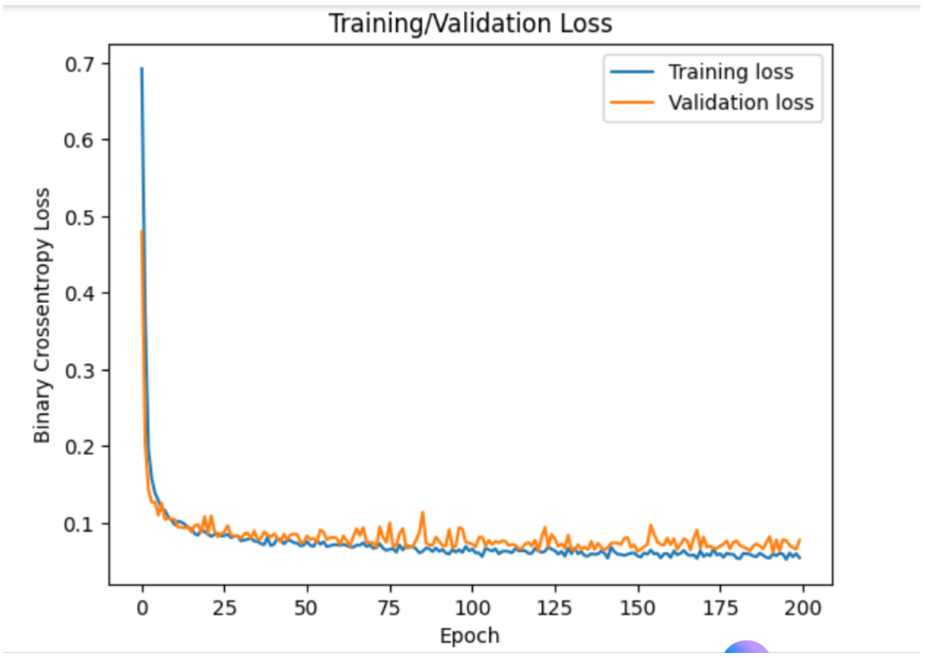
Then for the classification model, I do the same thing for the training set, test set and validation set.

Then I design a neural network with 2-neuron which takes in $x = (lon, lat)$ in the input layer, 32-neuron with the relu activation function in the first,second and third hidden layer, then the output layer has 1-neuron with sigmoid activation fuction outputs the probability as the hypothesis function.

I choose Binary cross entropy as the loss function.

Then to train the neural network, it will do a forward and back propagation for 200 epochs while using the mini-batch gradient descent for 32 data each time and Adaptive moment estimation as optimizer with learning rate 0.01.
Also, I give the loss function different weights to different class(0:1.0, 1:1.5) since the data are most of -999 value, the weights can give us a better result.

Next, we have the following result: Log Loss: 0.06681591573237361 .



Training/Validation Loss
Training loss
Validation loss
Binary Crossentropy Loss
Epoch
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
25
50
75
100
125
150
175
200