Neural Network Polynomial fitting

We want to use a neural network to predict the value of y based on three input variables x_1, x_2, x_3 . Instead of manually computing y using a formula, we train a neural network to learn the pattern from given data.

The actual function we are trying to approximate is:

$$y = x_1 - x_2^2 + 100x_3 + 4$$

This equation determines how y depends on x_1, x_2, x_3. However, we pretend that we don't know this equation and try to train a neural network to learn it.

Step 1: you can generate training data. Suppose now you have the observations:

$x_1 = -500:1:500$ (i.e., $x_1 = -500, -499, -498, ..., 500$, a row vector)

$x_2 = -500:1:500$

$x_3 = -500:1:500$

Stack x_1, x_2, x_3 into a 3×1001 observation matrix, and each column of the matrix corresponds to each sample data. Use one hidden layer to estimate y .

Step 2: Neural Network Learning.

we train a neural network with:

- Three inputs: x_1, x_2, x_3
- One hidden layer (with different numbers of neurons in different cases)
- One output node (to predict y)

Denote n as the number of hidden nodes in the hidden layer, and β is the portion of training data used to train the network.

Plot the predicted value of the network versus the target value for training data and test data separately, for the following three cases:

(a1) $n = 1, \beta = 0.7$

(a2) $n = 2, \beta = 0.7$

(a3) $n = 100, \beta = 0.7$

(a4) $n = 2, \beta = 0.3$

(a5) $n = 100, \beta = 0.3$

Discuss your findings from your results. Document all the required output figures and output matrix in a written report. Discuss about your results, e.g., why or why not your implementation works. Include the script at the end of the report.