Homework 5 Report
Name: Viprav Lipare
Student ID: 801288922
Homework Number: 5
Github Repository: https://github.com/vipravlipare/ECGR-5105-Intro-to-Machine-Learning

Problem 1A

In Problem 1, the lecture example for temperature prediction was modified by replacing the linear model from lecture with a nonlinear model. In this part the model was changed to include the quadratic term. Instead of predicting output as w1*t_u + b, we changed the forward equation into: w2 * (t_u)^2 + w1 * t_u + b. After changing the model the training loop had to be changed to fit the new quadratic version, but everything else stayed exactly the same. Adding the quadratic term changed the model's behavior, but it still struggled to converge properly with higher learning rates.

Problem 1B

The model was trained and the quadratic function was used. The models were trained over 5000 epochs over varying learning rates, 0.1, 0.01, 0.001, and 0.0001, and every 500 epochs were recorded. The dataset was the same as in lecture, and the inputs were normalized in the same way. The data was trained multiple times, but the difference is that the data was tested with multiple learning rates to see the differences in the learning rates and the results. Only the smallest learning rate (0.0001) produced stable results, showing that larger learning rates caused numerical instability (NaNs).

Problem 1C

For part C of problem 1 the best performing learning rate was taken from Problem 1.B. Then the loss values and results produced by the code given in the lecture were compared with the new model's values to show the difference in results, and to visually see the better model. The model was plotted with X-axis T_U (Unnormalized Temperature), Y-axis T_C (Celsius Temperature), and the Linear and Non-Linear models were plotted against each other. The linear model visually performed better than the non-linear model because the linear model was closer to all of the temperature points in the graph. This is also confirmed by the non-visual results as the linear model had a minimum loss value of 2.928, but the non-linear model had a minimum loss value of 3.8615.

<u>Problem 1B Results:</u>

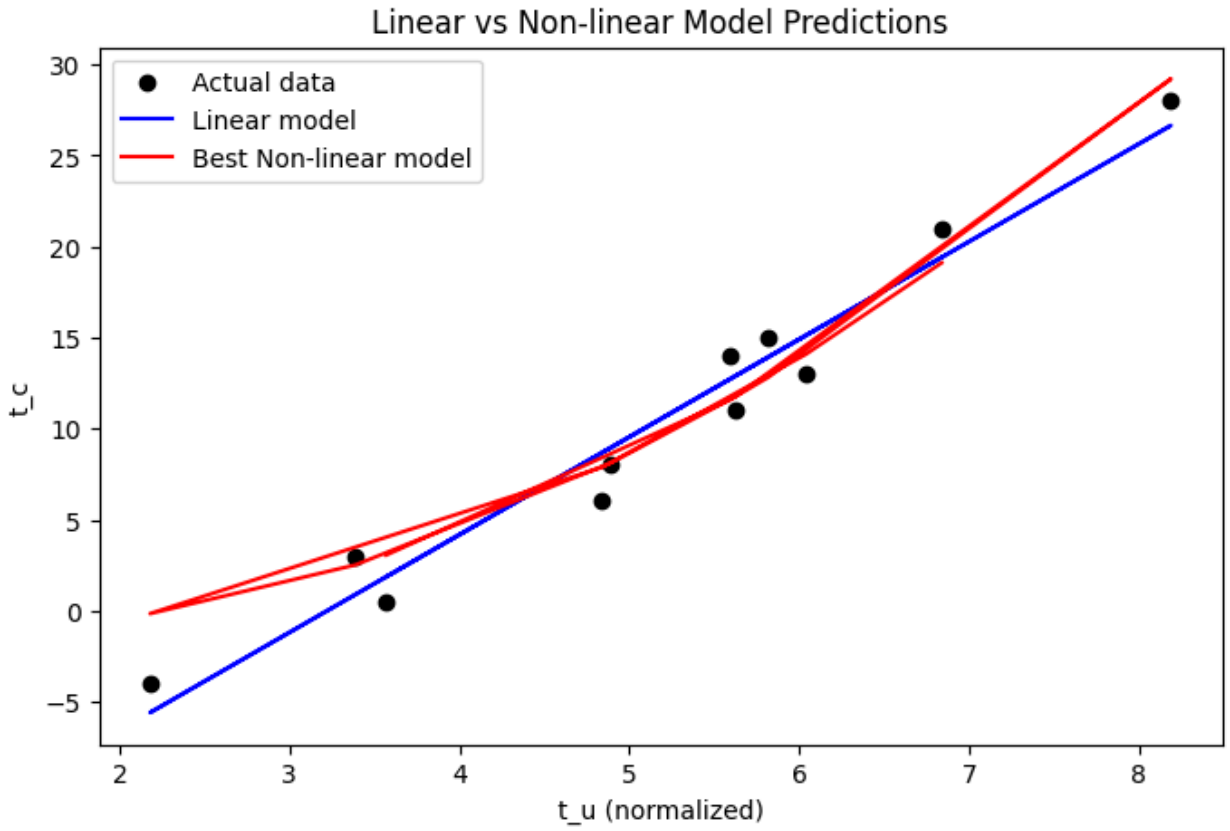
Training with learning rate = 0.1
Epoch 500, Loss nan
Epoch 1000, Loss nan
Epoch 1500, Loss nan
Epoch 2000, Loss nan
Epoch 2500, Loss nan

```
Epoch 3000, Loss nan
Epoch 3500, Loss nan
Epoch 4000, Loss nan
Epoch 4500, Loss nan
Epoch 5000, Loss nan
Final loss: nan
Final params: tensor([nan, nan, nan], requires_grad=True)

Training with learning rate = 0.01
Epoch 500, Loss nan
Epoch 1000, Loss nan
Epoch 1500, Loss nan
Epoch 2000, Loss nan
Epoch 2500, Loss nan
Epoch 3000, Loss nan
Epoch 3500, Loss nan
Epoch 4000, Loss nan
Epoch 4500, Loss nan
Epoch 5000, Loss nan
Final loss: nan
Final params: tensor([nan, nan, nan], requires_grad=True)

Training with learning rate = 0.001
Epoch 500, Loss nan
Epoch 1000, Loss nan
Epoch 1500, Loss nan
Epoch 2000, Loss nan
Epoch 2500, Loss nan
Epoch 3000, Loss nan
Epoch 3500, Loss nan
Epoch 4000, Loss nan
Epoch 4500, Loss nan
Epoch 5000, Loss nan
Final loss: nan
Final params: tensor([nan, nan, nan], requires_grad=True)

Training with learning rate = 0.0001
Epoch 500, Loss 10.708596
Epoch 1000, Loss 8.642083
Epoch 1500, Loss 7.171005
Epoch 2000, Loss 6.123477
Epoch 2500, Loss 5.377227
Epoch 3000, Loss 4.845285
Epoch 3500, Loss 4.465788
```

Epoch 4000, Loss 4.194724
Epoch 4500, Loss 4.000802
Epoch 5000, Loss 3.861744
Final loss: 3.8615105152130127
Final params: tensor([-0.8881, 0.5570, -0.8753], requires_grad=True)

Best nonlinear model learning rate: 0.0001
Best nonlinear final loss: 3.8615105152130127
Best params: tensor([-0.8881, 0.5570, -0.8753], grad_fn=<CloneBackward0>)

Problem 1C Results:
Linear model final loss: 2.9276480674743652
Best nonlinear model final loss: 3.8615105152130127



Problem 2A

For problem 2 the housing dataset was used, and only the area, bedrooms, bathrooms, stories, and parking were used to train the data. The inputs were normalized, and the data was split into 80% for training and 20% for validation. This formula following formula “U=W5*X5 + W4*X4 +

W3*X3 + W2*X2 + W1*X1 + B” was used as the formula within the model. The formula contains each input variable multiplied by a weight. The results were printed to see which weight value was the highest, and correspondingly which input variable is the most valuable. According to the results and a weight value of “2933135.8”, the Area input parameter is the most valuable input.

Problem 2B

The 6-input model was trained using 5000 epochs over varying learning rates, 0.1, 0.01, 0.001, and 0.0001, and every 500 epochs were recorded. The housing dataset was used in this problem, and the inputs were normalized and fit to training and validation. The data was trained multiple times, but the difference is that the data was tested with multiple learning rates to see the differences in the learning rates and the results. The lowest loss occurred during the 0.1 learning rate at Epoch 2000, and the lowest loss value was 2292614692864.0. The highest learning rate (0.1) converged fastest to the lowest loss, while lower learning rates produced higher losses, showing slower or incomplete convergence.

Results 2A:

Epoch 500 Loss= 1699869229056.0
Epoch 1000 Loss= 1556255735808.0
Epoch 1500 Loss= 1481826238464.0
Epoch 2000 Loss= 1436944433152.0
Validation loss: 2440182366208.0
Trained parameters [B-W5] = [2508056.5 2933135.8 1395469.2 2351054.8 1603754.9 1423087.1]

Results 2B:

Training with learning rate = 0.1
Epoch 500 Training Loss= 1358686322688.0 Validation Loss= 2313595125760.0
Epoch 1000 Training Loss= 1350310166528.0 Validation Loss= 2293964472320.0
Epoch 1500 Training Loss= 1350024036352.0 Validation Loss= 2292591099904.0
Epoch 2000 Training Loss= 1350009225216.0 Validation Loss= 2292614692864.0
Epoch 2500 Training Loss= 1350008176640.0 Validation Loss= 2292681539584.0
Epoch 3000 Training Loss= 1350008176640.0 Validation Loss= 2292710899712.0
Epoch 3500 Training Loss= 1350008045568.0 Validation Loss= 2292719812608.0
Epoch 4000 Training Loss= 1350008045568.0 Validation Loss= 2292721909760.0
Epoch 4500 Training Loss= 1350008176640.0 Validation Loss= 2292722434048.0
Epoch 5000 Training Loss= 1350008176640.0 Validation Loss= 2292722434048.0

Training with learning rate = 0.01
Epoch 500 Training Loss= 1699869229056.0 Validation Loss= 2915957211136.0
Epoch 1000 Training Loss= 1556255735808.0 Validation Loss= 2649620742144.0

Epoch 1500 Training Loss= 1481826238464.0 Validation Loss= 2515905806336.0
Epoch 2000 Training Loss= 1436944433152.0 Validation Loss= 2440182366208.0
Epoch 2500 Training Loss= 1408152895488.0 Validation Loss= 2393896648704.0
Epoch 3000 Training Loss= 1389229244416.0 Validation Loss= 2364096643072.0
Epoch 3500 Training Loss= 1376647643136.0 Validation Loss= 2344159281152.0
Epoch 4000 Training Loss= 1368219844608.0 Validation Loss= 2330412449792.0
Epoch 4500 Training Loss= 1362538266624.0 Validation Loss= 2320701849600.0
Epoch 5000 Training Loss= 1358683963392.0 Validation Loss= 2313703129088.0
Training with learning rate = 0.001
Epoch 500 Training Loss= 3540332052480.0 Validation Loss= 5707507695616.0
Epoch 1000 Training Loss= 2057299951616.0 Validation Loss= 3644702851072.0
Epoch 1500 Training Loss= 1917634740224.0 Validation Loss= 3342630125568.0
Epoch 2000 Training Loss= 1870883323904.0 Validation Loss= 3233899085824.0
Epoch 2500 Training Loss= 1834175954944.0 Validation Loss= 3161707511808.0
Epoch 3000 Training Loss= 1801548726272.0 Validation Loss= 3101782704128.0
Epoch 3500 Training Loss= 1772170379264.0 Validation Loss= 3048592113664.0
Epoch 4000 Training Loss= 1745609162752.0 Validation Loss= 3000403755008.0
Epoch 4500 Training Loss= 1721512230912.0 Validation Loss= 2956429361152.0
Epoch 5000 Training Loss= 1699577331712.0 Validation Loss= 2916155129856.0

Training with learning rate = 0.0001
Epoch 500 Training Loss= 19724295471104.0 Validation Loss= 24090727415808.0
Epoch 1000 Training Loss= 15514222985216.0 Validation Loss= 19448236867584.0
Epoch 1500 Training Loss= 12304728981504.0 Validation Loss= 15875069444096.0
Epoch 2000 Training Loss= 9857700724736.0 Validation Loss= 13121063223296.0
Epoch 2500 Training Loss= 7991701536768.0 Validation Loss= 10995033440256.0
Epoch 3000 Training Loss= 6568476672000.0 Validation Loss= 9350820134912.0
Epoch 3500 Training Loss= 5482659971072.0 Validation Loss= 8076610502656.0
Epoch 4000 Training Loss= 4653971406848.0 Validation Loss= 7086844936192.0
Epoch 4500 Training Loss= 4021239414784.0 Validation Loss= 6316007358464.0
Epoch 5000 Training Loss= 3537835130880.0 Validation Loss= 5713884610560.0

Problem 3A

For Problem 3, a fully connected neural network (NN) was built to predict housing prices using the same dataset and preprocessing from Problem 2.
For problem 3A the NN was designed with one hidden layer containing 8 nodes, using ReLU activation. The model was trained for 200 epochs with a learning rate of 0.01. Training and validation used an 80%-20% split of the normalized dataset. The final training time, training loss, and evaluation accuracy (R^2) after 200 epochs were recorded. The output produced the following results, training time of 0.333 seconds, Final training loss of 3.98*10^12, and the evaluation accuracy (R^2) was -0.3001. The time is as expected because the neural network

only has one layer, the loss was extremely high which is expected out of the housing dataset as the same dataset has produced higher loss values in previous homework assignments when using methods like linear regression and SGD. The evaluation accuracy is negative meaning that randomly predicting the price values would have been more effective, and the model did not learn anything significant from the training process using the neural network.

Results(3A):

Training time = 0.3331565856933594 seconds
Final Training Loss = 3985036279808.0
Evaluation Accuracy (R^2) = -0.30007660388946533

Problem 3B

For problem 3B the NN that was used for problem 3A was extended with two additional hidden layers, following a structure of Inputs → 8 → 8 → 8 → Output nodes, all using ReLU activations. The model was again trained for 200 epochs on the same normalized dataset. For problem 3B the two learning rates were used to analyze the differences in the learning rates impact on the NN. First a 0.01 learning rate was used, the same learning rate from problem 3A, but this learning rate gave NaN values, so a learning rate of 1e-9 was used to show much clearer values. The output produced the following results, training time of 0.277 seconds, Final training loss of 3.52*10^13, and the evaluation accuracy (R^2) was -4.96. The time is as expected, the loss was extremely high, the same housing dataset has produced higher loss values in previous homework assignments. The evaluation accuracy is negative meaning that the model did not learn anything significant from the training process using the neural network. The evaluation accuracy and final training loss were both worse when adding 2 additional hidden layers to the NN compared to the single hidden layer from problem 3A.
Overfitting occurs when the training loss is low and when the evaluation accuracy is bad. However, in this problem the evaluation accuracy is bad (negative values), but the training loss is extremely high. This means that there is no overfitting in this problem and that the results are bad. This means that the training did not cause overfitting but rather just bad training of the model. This is true for both learning rates 0.01 and 1e-9 as both learning rates have bad results.

Results(3B - 0.01):

Training time = 0.4250485897064209 seconds
Final Training Loss = nan
Evaluation Accuracy (R^2) = nan

Results(3B - 1e-9):

Training time = 0.2768397331237793 seconds
Final Training Loss = 25234774163456.0
Evaluation Accuracy (R^2) = -4.960936069488525