GitHub Code Repository: <https://github.com/pharmon0/Harmon_ECGR5105>

**Part 1a)**

**Develop a gradient decent training and evaluation code that predicts housing price based on the following input variables:**

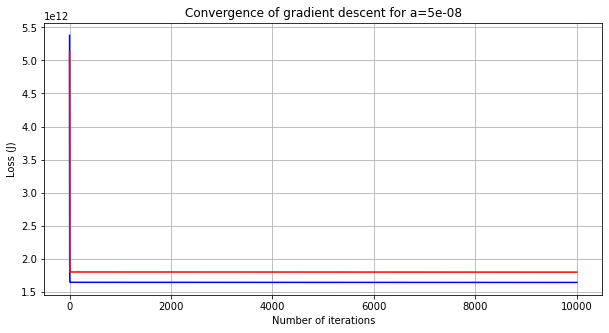
***[area, bedrooms, bathrooms, stories, parking]***

**Identify the best parameters for your linear regression model, based on the above input variables. Plot the training and validation losses (in a single graph, but two different lines). For the learning rate, explore different values between 0.1 and 0.01 (your choice). Initialize your parameters (thetas to zero). For the training iteration, choose what you believe fits the best.**

No value of the learning rate in the range of 0.1 to 0.01 produced a converging output for the gradient descent algorithm. The learning rate had a maximum value in the order of in order to achieve a non-divergent loss during learning.

Using a learning rate of , The following results were achieved after 10k iterations of the gradient descent:





These results are not good. The loss is enormous. Worth noting is that it may still be converging, as the graph is so highly scaled in the Y-axis that it is hard to tell, however it was reaching the maximum number of iterations that my computer could safely handle.

The final model parameters produced can be seen below:

Text

Description automatically generated

**Part 1b)**

**Develop a gradient decent training and evaluation code that predicts housing price based on the following input variables:**

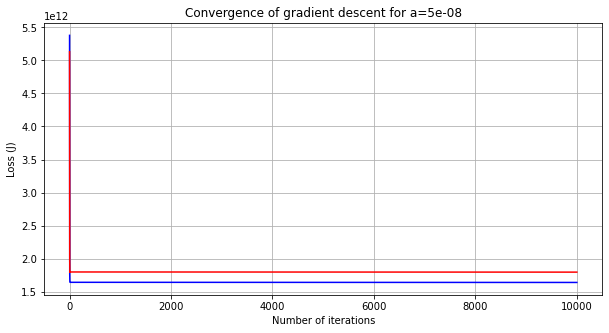
***[Area, bedrooms, bathrooms, stories, mainroad, guestroom, basement, hotwaterheating, airconditioning, parking, prefarea]***

**Identify the best parameters for your linear regression model, based on the above input variables. Plot the training and validation losses (in a single graph, but two different lines) over your training iteration. Compare your linear regression model against problem 1 a. For the learning rate, explore different values between 0.1 and 0.01 (your choice). Initialize your parameters (thetas to zero). For the training iteration, choose what you believe fits the best.**

Similarly to Part1a, no value of the learning rate in the range of 0.1 to 0.01 produced a converging output for the gradient descent algorithm. The learning rate had a maximum value in the order of in order to achieve a non-divergent loss during learning.

Using a learning rate of , The following results were achieved after 10k iterations of the gradient descent:





These results are not good. The loss is enormous. Worth noting is that it may still be converging, as the graph is so highly scaled in the Y-axis that it is hard to tell, however it was reaching the maximum number of iterations that my computer could safely handle.

The final model parameters produced can be seen below:

Text

Description automatically generated

**Part 2a)**

**Repeat problem 1 a, this time with input normalization and input standardization as part of your pre-processing logic. You need to perform two separate trainings for standardization and normalization.**

**Plot the training and validation losses for both training and validation set based on input standardization and input normalization. Compare your training accuracy between both scaling approaches as well as the baseline training in problem 1 a. Which input scaling achieves the best training? Explain your results.**

Adding the input scaling produced wonderful results. While the results without scaling were basically unusable, these results are quite good. Especially for the min/max normalized set. The loss is very low, and the validation set loss is roughly identical to the training set loss. The standardized set had an odd quirk of the validation set having less loss than the training set: a fortunate coincidence.

Text

Description automatically generated

A picture containing shoji, tiled

Description automatically generated

A picture containing text, shoji

Description automatically generated

From this data, the Min/Max normalization scaling is the best approach for this dataset because it both has the lowest loss, and the closest match between training and validation loss.

The final model parameters produced by both scaling methods can be seen below:

Table

Description automatically generated with medium confidence

**Part 2b)**

**Repeat problem 1 b, this time with input normalization and input standardization as part of your pre-processing logic. You need to perform two separate trainings for standardization and normalization.**

**Plot the training and validation losses for both training and validation set based on input standardization and input normalization. Compare your training accuracy between both scaling approaches as well as the baseline training in problem 1 b. Which input scaling achieves the best training? Explain your results.**

Like Part 2a, the input scaling had a tremendous effect on the efficacy of the training. The results of the training can be seen below.

Text

Description automatically generated

A picture containing shoji, crossword puzzle, tiled

Description automatically generated

A picture containing shoji

Description automatically generated

The data shows that, once again, the min/max normalization scaling produces the best loss characteristics for this dataset and produces the lowest difference between training loss and validation loss.

The final model parameters produced by both scaling methods can be seen below:

Table

Description automatically generated with medium confidence

**Part 3a)**

**Repeat problem 2 a, this time by adding parameters penalty to your loss function. Note that in this case, you need to modify the gradient decent logic for your training set, but you don’t need to change your loss for the evaluation set.**

**Plot your results (both training and evaluation losses) for the best input scaling approach (standardization or normalization). Explain your results and compare them against problem 2 a.**

**Text

Description automatically generated**

**A picture containing text, shoji, tiled

Description automatically generated**

**A picture containing text, shoji

Description automatically generated**

Once again, it seems that min/max normalization produces the best training results, and interestingly, the parameter penalization has resulted in an equivalent final loss value. This implies that the size of the parameters was not contributing to any training-validation error issues in Part 2a.

The final model parameters produced by both scaling methods can be seen below:

Text

Description automatically generated with medium confidence

**Part 3b)**

**Repeat problem 2 b, this time by adding parameters penalty to your loss function. Note that in this case, you need to modify the gradient decent logic for your training set, but you don’t need to change your loss for the evaluation set.**

**Plot your results (both training and evaluation losses) for the best input scaling approach (standardization or normalization). Explain your results and compare them against problem 2 b.**

Text

Description automatically generated

A picture containing text, shoji, crossword puzzle, tiled

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

Here we see that the min/max normalization once again produces better loss and training-validation error characteristics than the standardization. Additionally, the parameter penalization again seems not to have produced a significant effect on the end-loss of the training as compared to the non-penalized counterpart.

The final model parameters produced by both scaling methods can be seen below:

Table

Description automatically generated