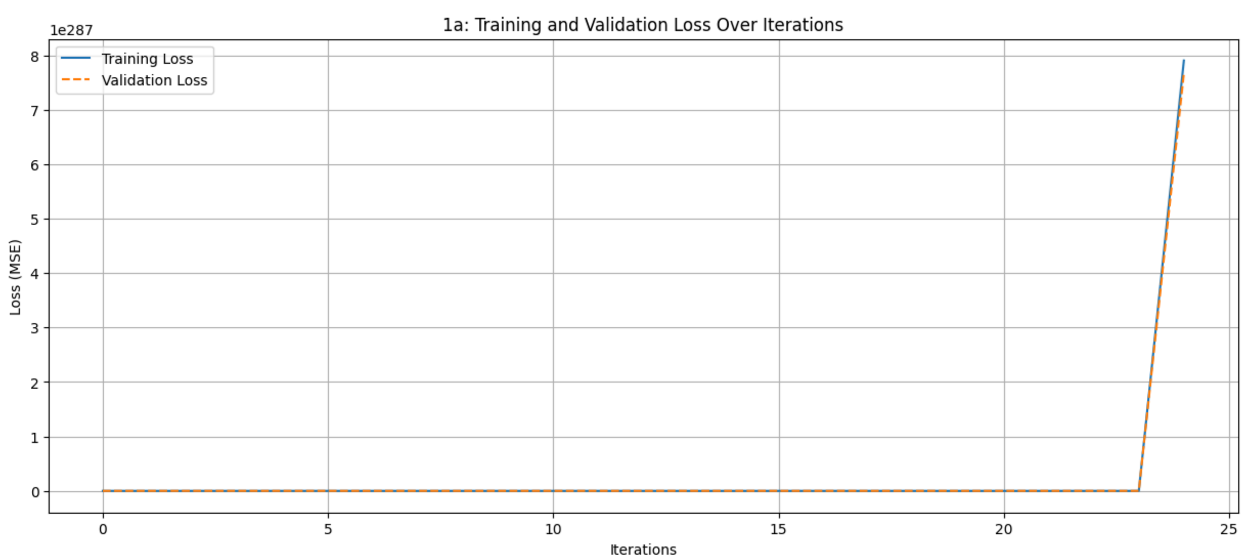
Nathon Tadeo
Student ID: 801265462
9/26/24
Homework 2
Github:https://github.com/nathon-tadeo/Intro-to-ML/blob/main/homework_2_intro_to_ml.ipynb

Problem 1 (30 points)

1.a) Develop a gradient decent training and evaluation code, from scratch, 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.

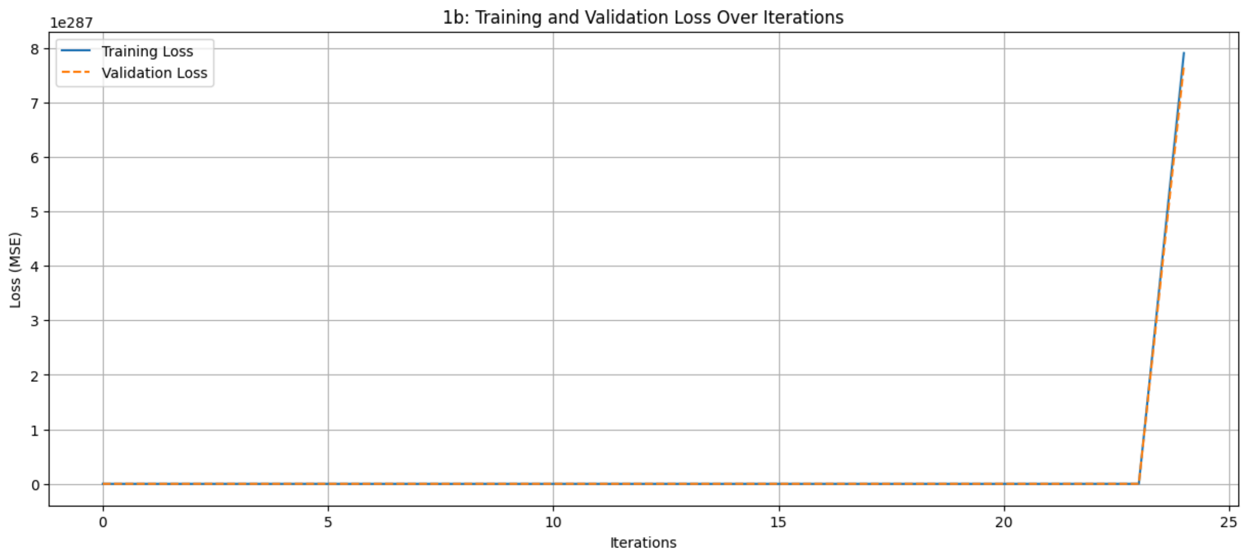
Plot the training and validation losses (in a single graph, but two different lines). For the learning rate, explore different values between 0.1 and 0.01 (your choice). Initialize your parameters (thetas to zero). For the training iteration, choose what you believe fits the best.

Using homework 1 as a model for the cost and gradient descent functions facilitated the process of finding the training loss and validation loss given the housing price data set. The data was split using “train_test_split” where the data was split into 80% data and 20% validation. Initially, the learning rate was 0.01 and the iterations were 1000, but once the gradient_decent was calculated and the loss was plotted, the code ran into an overflow error. The plot showed no convergence and displayed an infinitely increasing line. The iterations were reduced to 25 to get rid of the overflow error. There is likely no convergence in the graph due to the data lacking standardization and regularization. All plots are contained in “Tadeo_Nathon 801265462 (2)Homework2” pdf.”



1.b) Develop a gradient decent training and evaluation code, from scratch, that predicts housing price based on the following input variables: Area, bedrooms, bathrooms, stories, mainroad, guestroom, basement, hot water heating, 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.

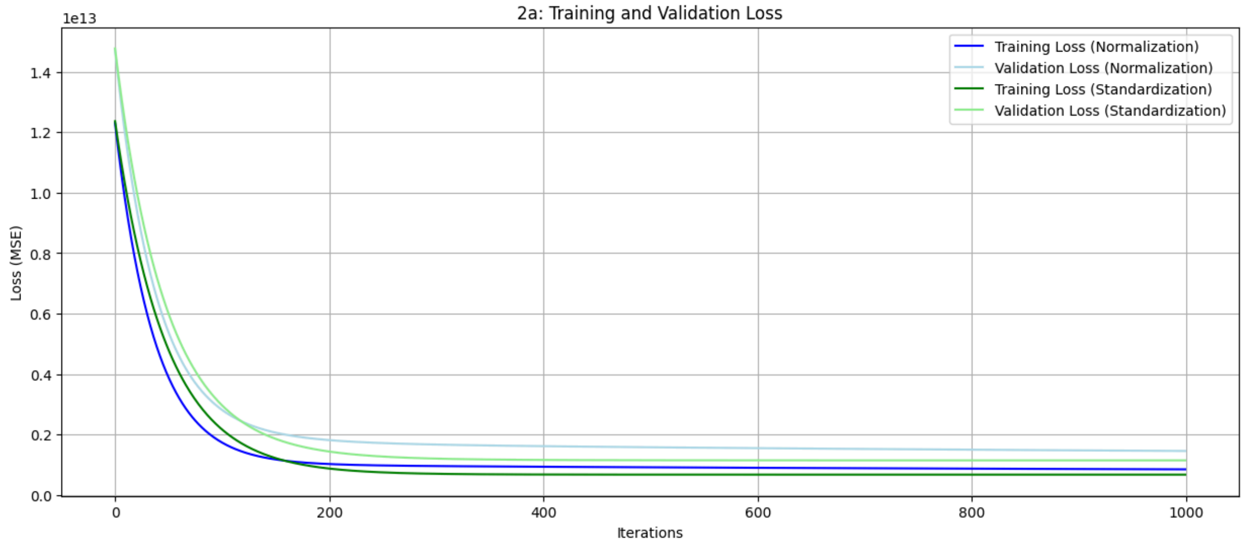
The process was the same as 1a, but the variable x contained all of the input variables within the housing price dataset. The learning rate was 0.01, and the training iterations had to be readjusted to 25 iterations to avoid the same overflow error that the previous model suffered from.



Problem 2 (30 points)

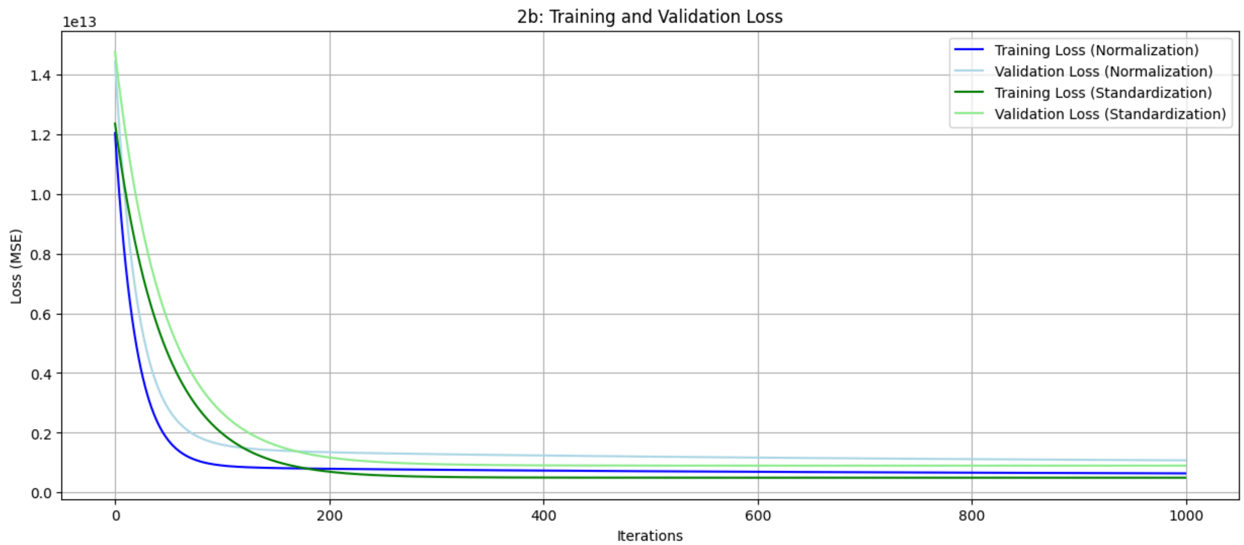
2.a) Repeat problem 1 a, this time with input normalization and input standardization as part of your pre-processing logic. You need to perform two separate trainings for standardization and normalization. **In both cases, you do not need to normalize the output!**
Plot the training and validation losses 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.

For this model, the inputs went through normalization and standardization. This was achieved by making separate functions and separate variables for the input. Standardization and normalization had separate trainings. The learning rate stayed at 0.01 for better comparison between the models, but the iterations were increased to 1000. In the graph, the green lines represented the loss with standardization scaled inputs and the blue lines represented the loss with normalization scaled inputs. Considering this model converges, this model's accuracy is significantly better than the model from problem 1a. The normalization scaling seems to have the least loss between 0 and 200 iterations, and the standardization scaling seems to have a lower loss than normalization when iterations are greater than 200.



2.b) 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. **In both cases, you do not need to normalize the output!**
Plot the training and validation losses for both training and validation sets based on input standardization and input normalization. Compare your training accuracy between both scaling approaches and the baseline training in problem 1b. Which input scaling achieves the best training? Explain your results.

This model contained the same steps for the normalization and standardization pre-processing of the data, except the input variables increased to all of the housing dataset This model's accuracy is significantly better than the model from problem 1b. The model followed the same trend of training and validation loss as the model from 2a. With the increased inputs, the plot showcases a greater difference in loss between normalization and standardization when the iterations are between 0 and 200. Between both models, normalization achieves the best training between 0 and 200 iterations and standardization achieves the best training when iterations are greater than 200.

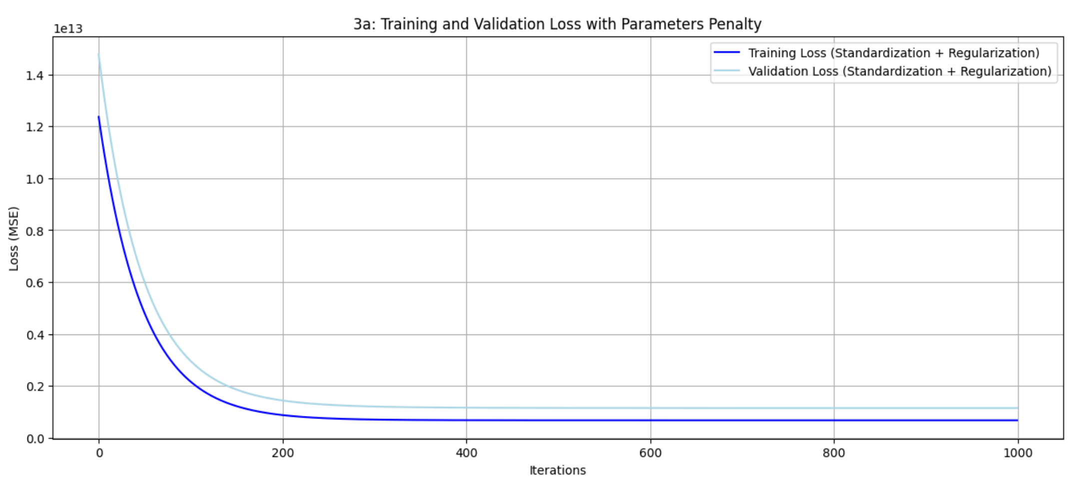


Problem 3 (40 points)

3.a) Repeat problem 2 a, this time by adding a 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.

The model from 2 was kept, but the cost and gradient descent functions were modified. These functions now account for parameter penalty using ridge regression. This adds regulation to the cost function to prevent overfitting by penalizing large values. The biggest change was adding the regularization term to the gradient decent and cost function. Overall, the results from this model and 2a were not very distinguishable. The models seem to match each other when accounting for parameter penalty. This could be due to the lamda value being too low (0.1)



3) b) Repeat problem 2 b, this time by adding a 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 2b.

Even after adding the additional inputs (Area, bedrooms, bathrooms, stories, mainroad, guestroom, basement, hot water heating, air conditioning, parking, prefarea), the parameter penalty did not seem to regularize the model by a significant amount. This model compared to 2b follows the same trends as 3a and 2a.