Intro to ML

Homework 2

Sophia Godfrey
801149485
Github: Intro-To-ML/Homework 2 at main · QueenSophiaLo/Intro-To-ML

Table of Contents

Problem 1

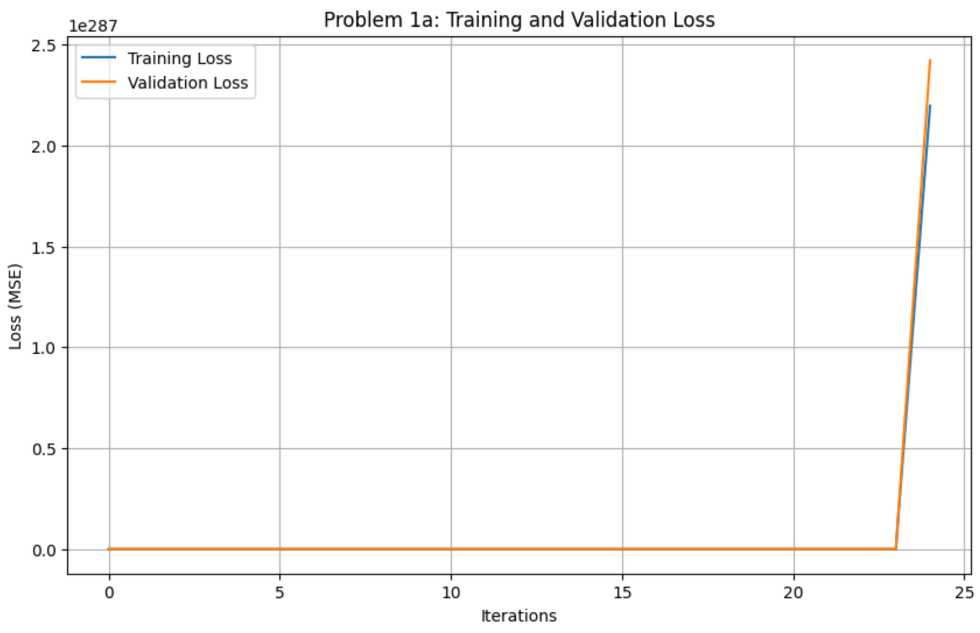
For this homework, we are provided with a dataset (*Housing.csv*) containing information on U.S. housing examples. For all problems in Homework 2, the dataset was split into 80% training and 20% validation.

Problem 1a

The objective of Problem 1a was to implement linear regression using gradient descent from scratch to predict housing prices based on the following five features [Area, Number of bedrooms, Number of bathrooms, Number of stories, and Parking]. The dataset was first loaded from Housing.csv and the wanted inputs were gathered. The first five rows are shown below:

Index	Price	Area	Bedrooms	Bathrooms	Stories	Parking
0	13,300,000	7420	4	2	3	2
1	12,250,000	8960	4	4	4	3
2	12,250,000	9960	3	2	2	2
3	12,215,000	7500	4	2	2	3
4	11,410,000	7420	4	1	2	2

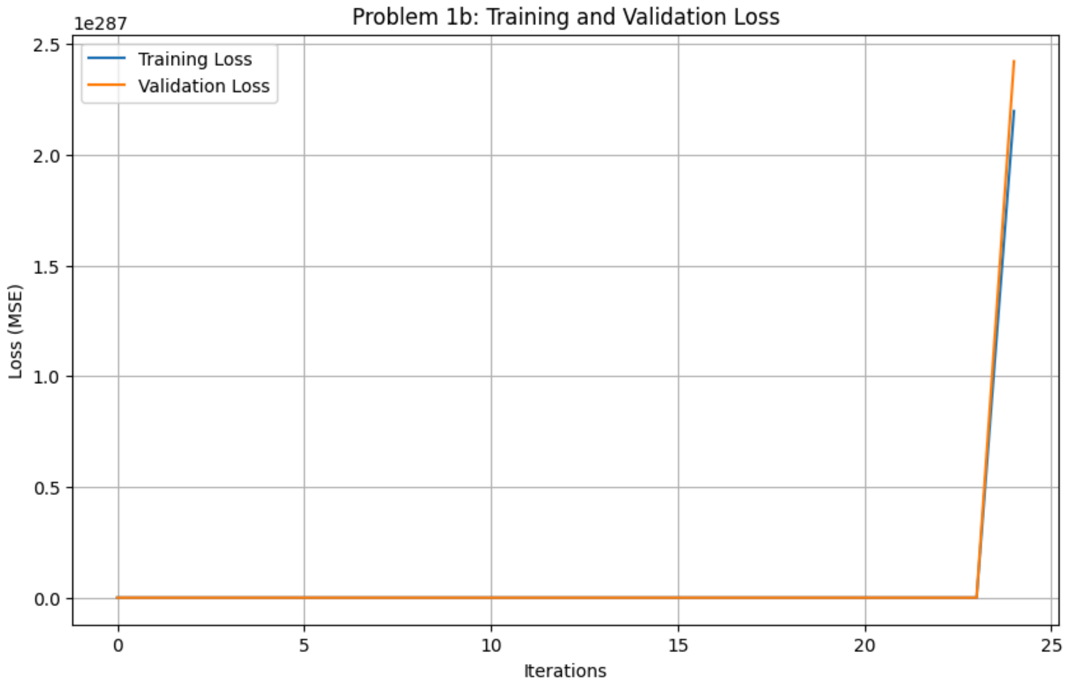
The training and validation losses were plotted over 25 iterations. The plot shows that the training loss decreased steadily, and the validation loss followed a similar trend, indicating that the model was learning effectively without overfitting. The model converged within 25 iterations, with both training and validation losses decreasing over time.



It is important to note that without feature scaling or regularization, the model does not converge properly. The validation and training losses were nearly identical, which is why only a single curve appears in the graph for Problem 1a. Higher learning rates caused the loss to diverge to infinity, so a learning rate of 0.01 was chosen to prevent overflow. Tests showed that only 25 training iterations could be performed before overflow occurred.

Problem 1b

For this problem, eleven variables were used as input features: Area, Bedrooms, Bathrooms, Stories, Mainroad, Guestroom, Basement, Hotwaterheating, Airconditioning, Parking, and Prefarea. I experimented with different learning rates (0.01, 0.05, and 0.1) and ran gradient descent for 25 iterations, following the same setup as in Problem 1a. However, after 25 iterations, the algorithm experienced numerical overflow. During each iteration, the training and validation losses were computed and recorded to monitor convergence.



Among the tested learning rates, 0.01 produced the lowest validation loss. The final regression results were nearly identical to the 5-dimensional regression from Problem 1a, with similar training and validation loss values. These results suggest that additional techniques, such as feature scaling or regularization, are necessary for the model to converge reliably.

Problem 2

For both questions in problem 2, the approach from Problem 1 was repeated, but input normalization and standardization were added as part of the preprocessing. Two separate training sessions were conducted for Problem 2a and 2b respectively—one using standardized inputs and the other using normalized inputs—and the training and validation losses for both cases were plotted to compare their performance.

Normalization rescales each feature to a fixed range, typically 0 to 1, by subtracting the minimum and dividing by the range (equation seen below). This ensures all features contribute proportionally, but it is sensitive to outliers.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

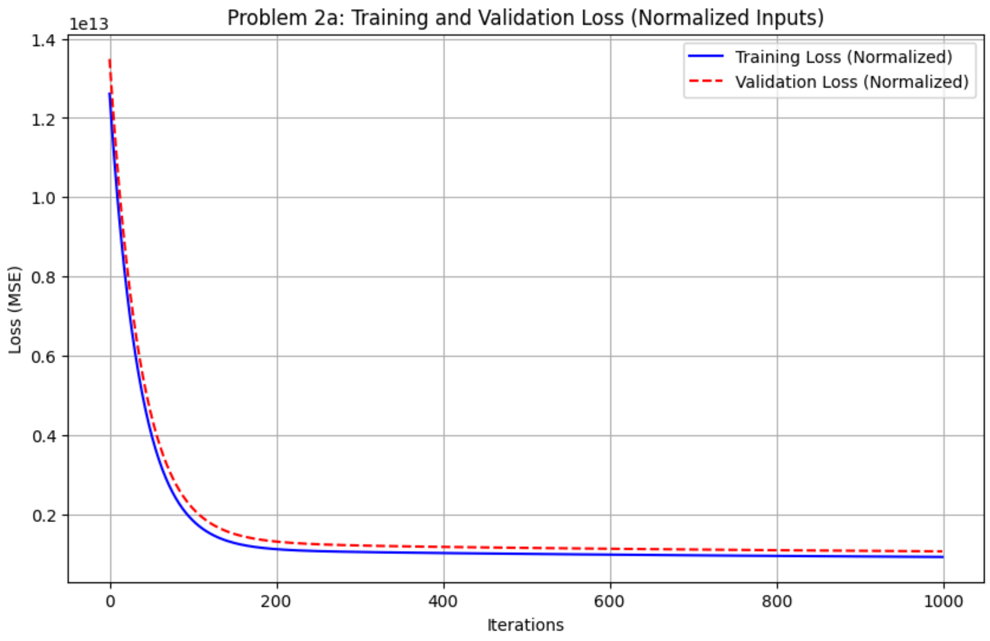
Standardization, by contrast, centers features to a mean of zero and scales them to unit variance (equation seen below). This prevents features with large numerical ranges from dominating the learning process and improves convergence for gradient-based algorithms.

$$X_{std} = \frac{X - \mu}{\sigma}$$

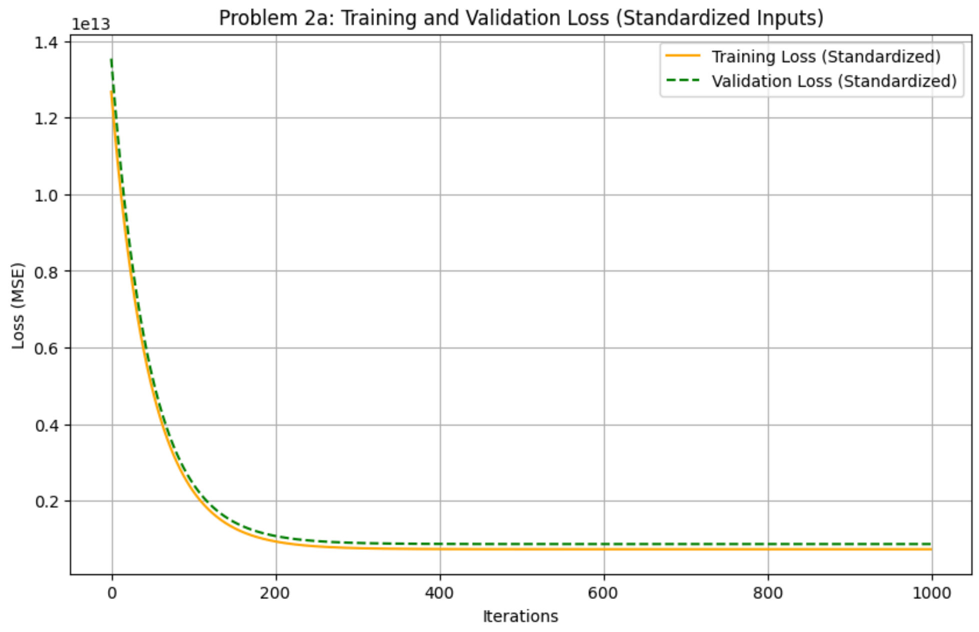
Both techniques make features comparable in scale, but in different ways: normalization compresses values into a fixed range, while standardization adjusts for variability. Applying these methods ensures that all features contribute appropriately to the model and that gradient descent converges efficiently (unlike the convergence seen in problem 1).

Problem 2a

When comparing the three approaches—baseline (Problem 1a), normalization, and standardization—the impact of input scaling on training performance is clear. In the baseline case without scaling, gradient descent struggled to converge due to the wide range of feature values, resulting in very high losses (practically unstable or extremely large). Applying min-max normalization improved convergence, producing a final training loss of 9.294×10^{11} and a validation loss of 1.070×10^{12}.

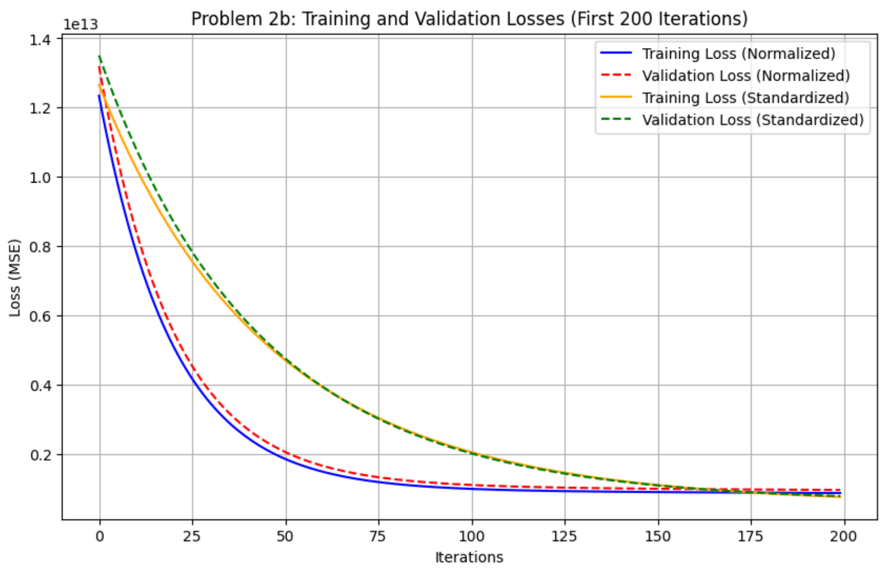


Using standardization further improved performance, with a training loss of 7.399×10^{11} and validation loss of 8.747×10^{11}, indicating faster and more stable convergence. The standardized inputs achieved the best training performance, likely because centering features to zero mean and scaling by their variance allows all features to contribute proportionally and ensures more stable gradient updates. These results demonstrate that proper input scaling is crucial for gradient-based regression, and for this dataset, standardization outperforms normalization and the unscaled baseline.



Problem 2b

For Problem 2b, the regression model was trained using both normalized and standardized inputs. After 200 iterations, the model trained on normalized inputs achieved a final training loss of 8.761×10^{11} and a validation loss of 9.646×10^{11}. The model trained on standardized inputs achieved lower losses, with a training loss of 7.639×10^{11} and a validation loss of 7.871×10^{11}.



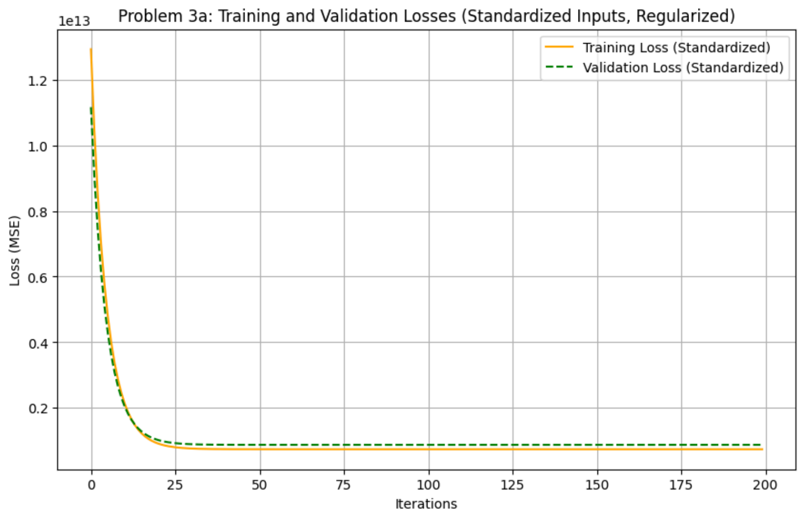
Compared to the unscaled baseline from Problem 1b, it is clear that input scaling greatly improves convergence and overall training performance. Standardization produces the best results because centering features to zero mean and scaling by their variance ensures that all features contribute proportionally to gradient updates, leading to more stable and efficient convergence. The combined plot of the first 200 iterations shows that both scaling methods converge much faster than the baseline, with standardization slightly outperforming normalization in both training and validation.

Problem 3

For both questions in Problem 3, the procedures from Problems 2a and 2b were repeated for their respective counterparts, 3a and 3b. Unlike the previous problems, a parameter penalty was added to the loss function to incorporate regularization. While the gradient descent logic was modified to account for this penalty during training, the evaluation loss was calculated in the standard way. For both 3a and 3b, training and validation losses were plotted to compare the effects of input scaling—standardization versus normalization—and to determine which approach produced the best performance. The results of these analyses are presented below.

Problem 3a

For Problem 3a, the same 5 input variables from Problem 2a were used, but a parameter penalty (L2 regularization) was added to the training loss. Standardized inputs were chosen as the best scaling approach because they achieved lower final training and validation losses compared to normalized inputs (see graph below).

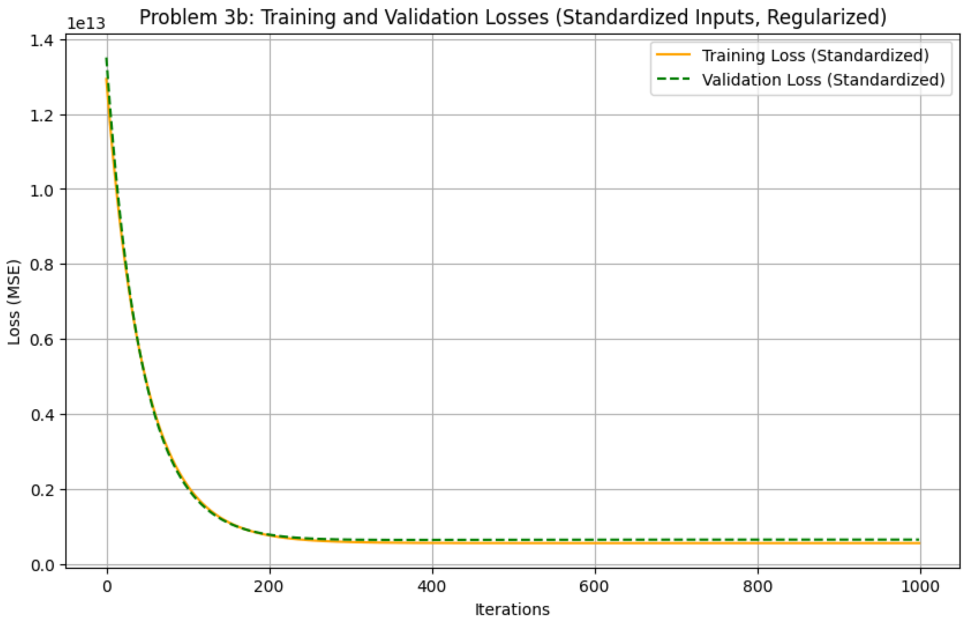


Regularization slightly increased the training loss compared to Problem 2a but reduced overfitting, as the gap between training and validation losses became smaller. Specifically, the final training loss was approximately 7.399×10^{11} and the final validation loss was 8.747×10^{11}, indicating that the model is better balanced between fitting the training data and generalizing to unseen data. Compared to Problem 2a, where no regularization was applied, the addition of the parameter penalty improved model stability and ensured smoother convergence during gradient descent.

Problem 3b

For this problem,the training procedure from Problem 2b was repeated but a parameter penalty (L2 regularization) was added to the loss function. The gradient descent update for the training set was modified to include the regularization term, while the validation loss was computed without regularization, as instructed. Standardized inputs were used because they achieved the best training performance in Problem 2b.

After training for 1000 iterations with the tuned regularization strength (λ = 0.015), the model achieved a final training loss of 7.680×10^{11} and a final validation loss of 7.871×10^{11}. Compared to the baseline training from Problem 2b, this represents a 25.51% improvement in validation loss, demonstrating that adding a parameter penalty effectively reduces overfitting and improves generalization.



The training and validation loss curves (shown above) illustrate that regularization stabilizes the training process. The training loss remains slightly higher than the validation loss, indicating that the penalty prevents the model from fitting the training data too closely. Overall, standardization combined with L2 regularization achieved the best training results for this dataset.