CS589: Machine Learning: Homework 4

Vishnu Vardhan Reddy Bheem Reddy

October 23, 2025

Question 1: The Constant Regressor

We are given the model $f(x) = b$ and the mean squared error (MSE) loss function:

$$L(b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - b)^2$$

To find the value of b that minimizes this loss, we take the derivative of $L(b)$ with respect to b and set it to zero.

$$\begin{aligned}
\frac{dL}{db} &= \frac{d}{db}\left[\frac{1}{N}\sum_{i=1}^{N}(y_i - b)^2\right] \\
&= \frac{1}{N}\sum_{i=1}^{N}\frac{d}{db}(y_i - b)^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}2(y_i - b)\cdot(-1) \quad \text{(using the chain rule)} \\
&= -\frac{2}{N}\sum_{i=1}^{N}(y_i - b)
\end{aligned}$$

Now, we set the derivative to zero and solve for $\hat{b}$:

$$\begin{aligned}
0 &= -\frac{2}{N}\sum_{i=1}^{N}(y_i - \hat{b}) \\
0 &= \sum_{i=1}^{N}(y_i - \hat{b}) \\
0 &= \left(\sum_{i=1}^{N} y_i\right) - \left(\sum_{i=1}^{N} \hat{b}\right) \\
0 &= \left(\sum_{i=1}^{N} y_i\right) - N\hat{b} \\
N\hat{b} &= \sum_{i=1}^{N} y_i \\
\hat{b} &= \frac{1}{N}\sum_{i=1}^{N} y_i
\end{aligned}$$

This shows that the $\hat{b}$ is the mean of the target values y_i.

Question 2: Ridge Regression Derivation

The optimization problem is:

$$\hat{w} = \arg\min_{w} \left(\left(\frac{1}{N}\sum_{i=1}^{N}(y_i - w^\top x_i)^2 \right) + \lambda \|w\|_2^2 \right)$$

a. Objective Function in Matrix Form

We are given the output vector Y ($N \times 1$), the input matrix X ($N \times D$), and the weight vector w ($D \times 1$). The vector of errors is $Y - Xw$. The sum of squared errors is $\|Y - Xw\|_2^2 = (Y - Xw)^\top (Y - Xw)$. The L_2-regularizer is $\lambda\|w\|_2^2 = \lambda w^\top w$. The full objective function $J(w)$ is:

$$J(w) = \frac{1}{N}(Y - Xw)^\top (Y - Xw) + \lambda w^\top w$$

b. Gradient of the Objective Function

First, we expand the objective:

$$J(w) = \frac{1}{N}(Y^\top Y - 2w^\top X^\top Y + w^\top X^\top X w) + \lambda w^\top w$$

Taking the gradient $\nabla_w J(w)$ term by term, using the provided matrix calculus results:

$$\begin{aligned}
\nabla_w J(w) &= \frac{1}{N}\nabla_w(Y^\top Y - 2w^\top X^\top Y + w^\top X^\top X w) + \nabla_w(\lambda w^\top w) \\
&= \frac{1}{N}(0 - 2X^\top Y + (X^\top X + (X^\top X)^\top)w) + \lambda(2Iw) \\
&= \frac{1}{N}(-2X^\top Y + 2X^\top X w) + 2\lambda w \quad \text{(since } X^\top X \text{ is symmetric)} \\
\nabla_w J(w) &= -\frac{2}{N}X^\top Y + \frac{2}{N}X^\top X w + 2\lambda w
\end{aligned}$$

c. Solving for the Optimal Parameters

Set the gradient to zero to find the optimal $\hat{w}$:

$$0 = -\frac{2}{N}X^\top Y + \frac{2}{N}X^\top X\hat{w} + 2\lambda\hat{w}$$

Multiply the entire equation by $N/2$:

$$0 = -X^\top Y + X^\top X\hat{w} + N\lambda\hat{w}$$

Isolate terms with $\hat{w}$:

$$X^\top X\hat{w} + N\lambda I\hat{w} = X^\top Y$$

Factor $\hat{w}$ out:

$$(X^\top X + N\lambda I)\hat{w} = X^\top Y$$

Then, pre-multiply by the inverse:

$$\hat{w} = (X^\top X + N\lambda I)^{-1}X^\top Y$$

Question 3: Ridge Regression Implementation

a. Implement predict

N/A.

b. Implement fit

N/A.

c. $\lambda = 0$ Check

The MSEs for our Ridge Regressor with $\lambda = 0$ are compared to scikit-learn's OLS Linear Regression. The results match, confirming correctness.

- **Our RidgeRegressor ($\lambda = 0$):**
  - Train MSE: 9.0189
  - Valid MSE: 10.9152

  - Test MSE: 8.1723

- **sklearn.linear_model.LinearRegression:**
  - Train MSE: 9.0189
  - Valid MSE: 10.9152
  - Test MSE: 8.1723

d. $\lambda \to \infty$ Check

The MSEs for our Ridge Regressor with a large lambda are compared to the Constant Regressor. The results are almost identical, confirming the correctness.

- **Our RidgeRegressor ($\lambda = 10^5$):**
  - Train MSE: 18.1534
  - Valid MSE: 23.5467
  - Test MSE: 17.6016
- **Constant Regressor:**
  - Train MSE: 18.1542
  - Valid MSE: 23.5476
  - Test MSE: 17.6024

e. Hyper-parameter Optimization

The following graph shows the training and validation MSE as a function of the regularization strength λ on a log scale.

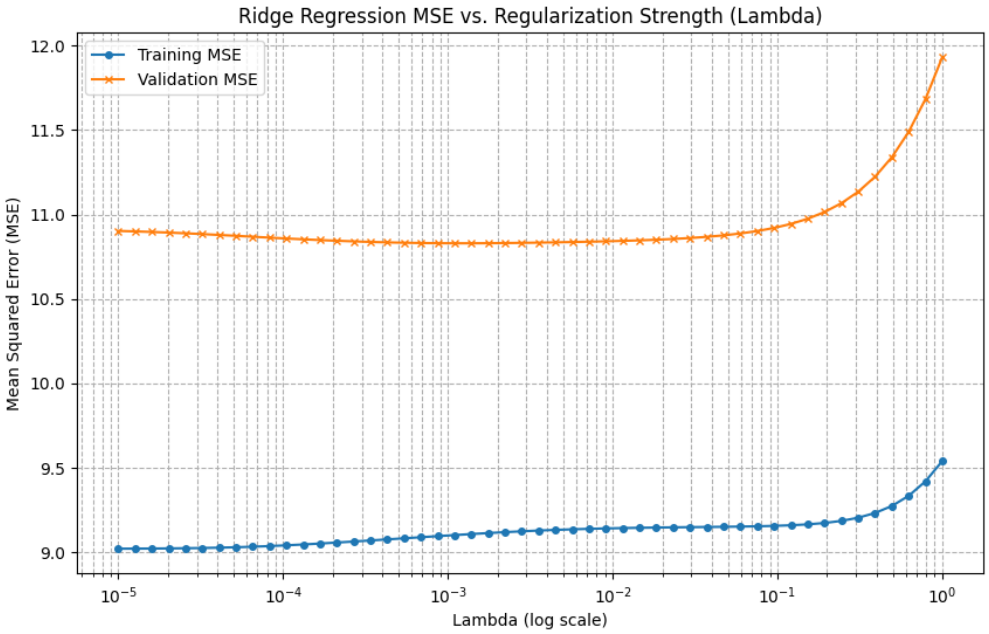

Figure 1: Ridge Regression MSE vs. Regularization Strength (λ).

f. Optimal λ and Final Results

Based on the above experiment, the optimal value of λ is the one that minimized the validation MSE. The final results are below.

- **Optimal Lambda (λ^*):** 0.001099
- **Train MSE (at λ^*):** 9.1002
- **Validation MSE (at λ^*):** 10.8294
- **Test MSE (at λ^*):** 7.8752

Question 4: Regularization Paths

a. Ridge Regularization Path

The following graph shows the regularization paths for the eight feature weights using Ridge (L2) regression as the regularization strength α increases. The weights shrink towards zero but do not become exactly zero.

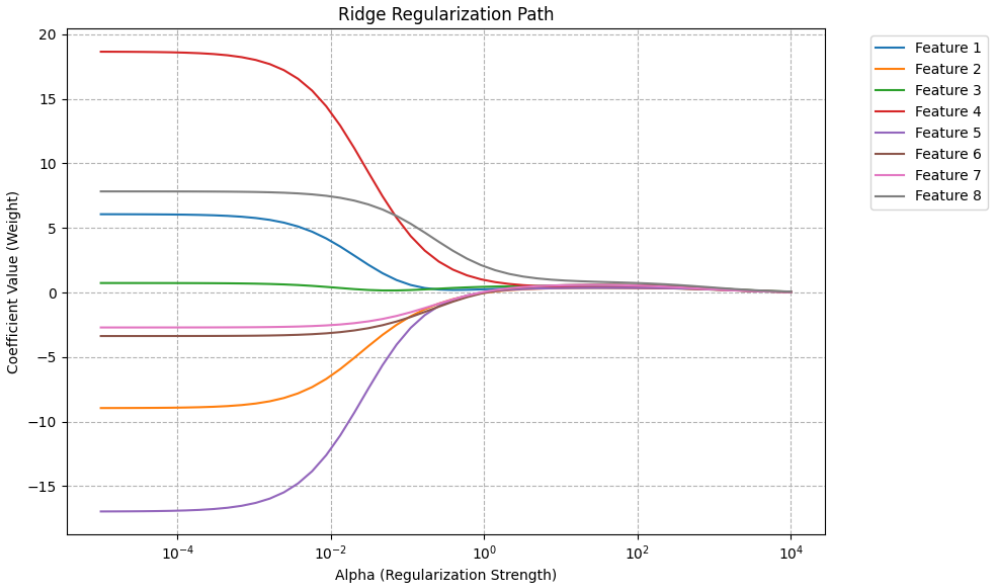

Figure 2: Regularization paths for Ridge (L2) regression coefficients vs. α.

b. Lasso Regularization Path

The following graph shows the regularization paths for the eight feature weights using Lasso (L1) regression as the regularization strength α increases. Many coefficients are forced to exactly zero.

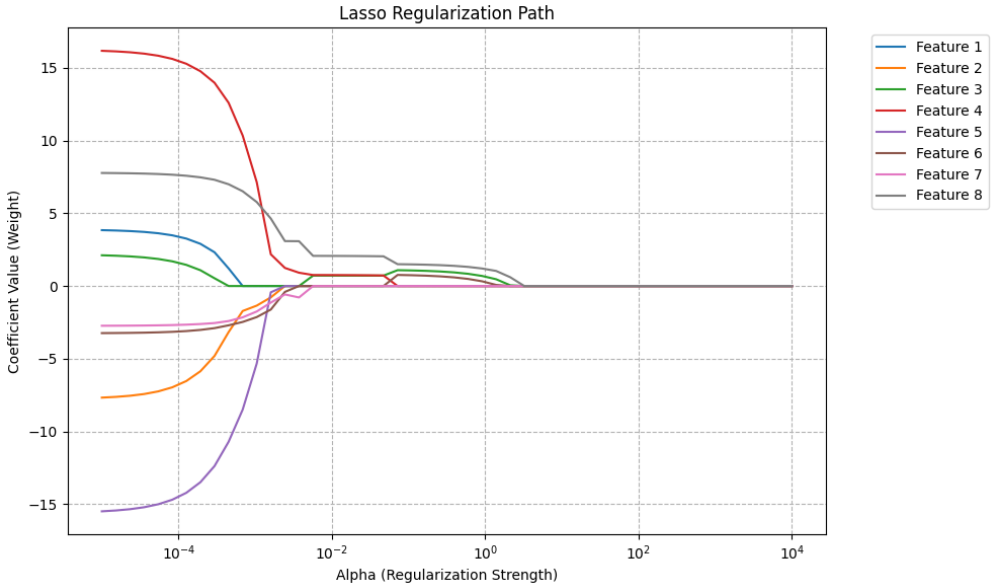

Figure 3: Regularization paths for Lasso (L1) regression coefficients vs. α.

c. Comparison of Regularization Paths

When comparing the graphs, some differences are clear:

In the **Lasso** regularization path plot (b), many of the feature weights are forced to be **exactly zero** as the regularization strength α increases.

In contrast, the **Ridge** regularization path plot (a) shows the weights shrinking *towards* zero, but they do not become exactly zero, even for large values of α.

- **Lasso** The L1 penalty promotes sparse solutions. This means it performs automatic feature selection by setting the coefficients of less important features to exactly zero.
- **Ridge** The L2 penalty shrinks all coefficients proportionally to penalize large weights but does not force them to be exactly zero. It retains all features in the final model.

Question 5: Bagging

a. Train-Test Split

After splitting the dataset (80% train, 20% test, with `random_state=0`), the sizes of the resulting sets are:

- **Training set size:** 16,512
- **Test set size:** 4,128

b. Resample Function

After applying the ‘resample‘ function to the training data set

- **Original training cases not included:** 36.54%

c. Implement BaggedTrees fit

N/A.

d. Implement BaggedTrees predict

N/A.

e. Learning Experiment

The following plot shows the Training MSE and Test MSE as a function of the ensemble size, K.

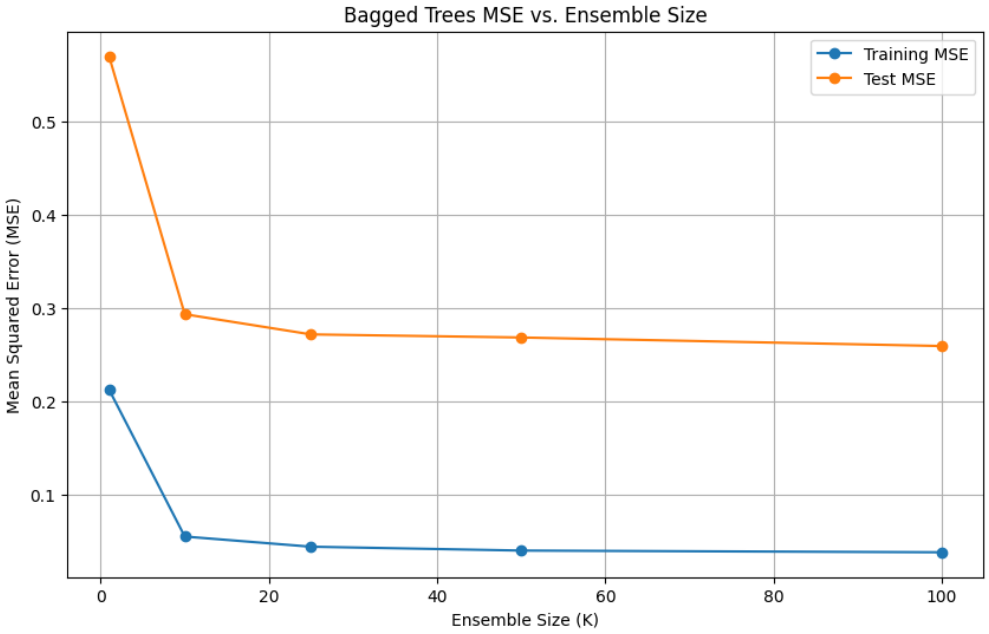

Figure 4: Bagged Trees MSE vs. Ensemble Size (K).

f. Analysis

- **Single Tree vs. Ensemble:** A single decision tree ($K = 1$) performs significantly worse than an ensemble. The single tree has a Test MSE of 0.5701, while the ensemble with $K = 100$ has a much lower Test MSE of 0.2592. The single tree, with a max depth of 20, is highly overfit to the training data. Using an ensemble (bagging) averages out this high variance, leading to a much better, more generalizable model.

- **Overfitting as K Increases:** No, we do not need to worry about overfitting as the value of K increases. As seen in the plot from Figure 4, the Test MSE consistently decreases as K gets larger. Bagging is an ensemble method that reduces variance. Adding more trees to the ensemble only serves to stabilize the average, and it does not increase the overall model complexity in a way that leads to overfitting.

Question 6: Generative AI Use

I used Gen AI tools primarily to help with boilerplate code and debugging. And for help with NumPy syntax formatting and specific method names.