Exercise 1. (Linear regression)

You are given a data matrix $X \in \mathbb{R}^{N\times(d+1)}$:

$$X := \begin{pmatrix} 1 & x_1^1 & x_2^1 & \dots & x_d^1 \\ 1 & x_1^2 & x_2^2 & \dots & x_d^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^N & x_2^N & \dots & x_d^N \end{pmatrix}.$$

We also define a weight vector $w \in \mathbb{R}^{d+1}$:

$$w := \begin{pmatrix} b \\ w_1 \\ \vdots \\ w_d \end{pmatrix},$$

where b is the bias, and $w_1, \dots, w_d$ are the weights corresponding to the features.

1. Verify that $J(w) = \frac{1}{N}\sum_{i=1}^{N}(w^\top x^i - y^i)^2$ can be equivalently written as $J(w) = \frac{1}{N}(Xw - y)^\top(Xw - y)$.

   Solution:

   Step 1. Expand the quadratic form:

   $$(Xw - y)^\top(Xw - y) = (Xw)^\top(Xw) - (Xw)^\top y - y^\top(Xw) + y^\top y.$$

   Step 2. Rewrite each term:

   $$\begin{aligned} (Xw)^\top(Xw) &= w^\top X^\top X w, \\ (Xw)^\top y &= w^\top X^\top y, \\ y^\top(Xw) &= y^\top X w. \end{aligned}$$

   Since the last two terms are scalars and are transposes of each other, they are equal:

   $$w^\top X^\top y = (w^\top X^\top y)^\top = y^\top X w.$$

   Step 3. Simplify:

   $$(Xw - y)^\top(Xw - y) = w^\top X^\top X w - 2\, w^\top X^\top y + y^\top y.$$

   Or equivalently (writing the middle term with $y^\top Xw$):

   $$(Xw - y)^\top(Xw - y) = w^\top X^\top X w - 2\, y^\top X w + y^\top y.$$

   Step 4. Divide by N:

   $$J(w) = \frac{1}{N}\big(w^\top X^\top X w - 2\, y^\top X w + y^\top y\big).$$

Step 5. Match with scalar summation form:

$$w^\top X^\top X w = \sum_{i=1}^{N} (w^\top x^i)^2, \quad y^\top X w = \sum_{i=1}^{N} y^i (w^\top x^i), \quad y^\top y = \sum_{i=1}^{N} (y^i)^2,$$

so

$$\frac{1}{N} \sum_{i=1}^{N} \big((w^\top x^i)^2 - 2 y^i w^\top x^i + (y^i)^2 \big) = \frac{1}{N} \sum_{i=1}^{N} (w^\top x^i - y^i)^2.$$

2. Now consider the regularized loss function $J(w) = \frac{1}{N} \sum_{i=1}^{N} (w^\top x^i - y^i)^2 + \lambda(w_1^2 + \ldots + w_d^2)$. Find the gradient and Hessian of the regularized loss function with respect to the model parameters.

Solution:

Component form of the gradient.

For the bias parameter b (the first component):

$$\frac{\partial J}{\partial b} = \frac{2}{N} \sum_{i=1}^{N} (w^\top x^i - y^i).$$

For the weights w_j, $j = 1, \ldots, d$:

$$\frac{\partial J}{\partial w_j} = \frac{2}{N} \sum_{i=1}^{N} (w^\top x^i - y^i)\, x_j^i \; + \; 2\lambda w_j.$$

Vector form of the gradient.

The regularized loss can be equivalently written as:

$$J(w) \; = \; \frac{1}{N} (Xw - y)^\top (Xw - y) \; + \; \lambda\, w^\top R w,$$

where $R = \mathrm{diag}(0, 1, \ldots, 1)$ ensures that the bias b is not regularized. Then, the gradient of the regularized loss in vector form is given by:

$$\nabla_w J(w) = \frac{2}{N} X^\top (Xw - y) + 2\lambda R w.$$

It can equivalently be written as:

$$\nabla_w J(w) = \frac{2}{N} X^\top (Xw - y) + 2\lambda \begin{pmatrix} 0 \\ w_1 \\ \vdots \\ w_d \end{pmatrix}.$$

Vector form of the Hessian.

$$\begin{aligned}\nabla_w^2 J(w) &= \nabla_w(\nabla_w J(w)) \\ &= \nabla_w \left(\frac{2}{N} X^\top (Xw - y) + 2\lambda \begin{pmatrix} 0 \\ w_1 \\ \vdots \\ w_d \end{pmatrix} \right) \\ &= \frac{2}{N} X^\top X + 2\lambda \begin{pmatrix} 0 & \mathbf{0_d}^T \\ \mathbf{0_d} & I_{d\times d} \end{pmatrix},\end{aligned}$$

where $I_{d\times d}$ is the identity matrix of size $d \times d$ and $\mathbf{0_d}$ is zero vector of size $d \times 1$.

3. Now, we consider unregulated loss function denoted by $J(w) = \frac{1}{N}(Xw - y)^\top (Xw - y)$. Assume $X^\top X$ is invertible. Find the optimal weight vector $w^\star$ that minimizes $J(w)$, i.e. $w^\star = \arg\min_w J(w)$.

   Solution: The loss function $J(w)$ is a quadratic function in w. Its Hessian is

   $$\nabla_w^2 J(w) = \tfrac{2}{N} X^\top X,$$

   which is positive definite. Therefore, $J(w)$ is convex. This means that any stationary point we find is guaranteed to be a *global minimizer*. Therefore, we need to find w^* such that

   $$\nabla_w J(w^*) = \frac{2}{N} X^\top (Xw^* - y) = 0 \Longrightarrow \quad X^\top X w^* = X^\top y.$$

   Therefore, $w^* = (X^\top X)^{-1} X^\top y$.

   Remark: If $X^\top X$ is not invertible (i.e. the columns of X are linearly dependent), the minimizer may not be unique. In this case, one can use the Moore–Penrose pseudoinverse to obtain the minimum-norm solution:

   $$w^\star = (X^\top X)^\dagger X^\top y,$$

   where $(X^\top X)^\dagger$ denotes the pseudoinverse of $X^\top X$.

4. Now we consider k feature functions $\{\phi_i(x)\}_{i=1}^k$, where $\phi_i : \mathbb{R}^d \to \mathbb{R}$. What is the corresponding data matrix?

   Solution: For each original data point $x^j \in \mathbb{R}^d$ with $j = 1, \ldots, N$, we apply the feature functions to obtain a new representation

   $$\Phi(x^j) := \big(\phi_1(x^j),\, \phi_2(x^j),\, \ldots,\, \phi_k(x^j)\big).$$

   The corresponding data matrix is then

   $$X := \begin{pmatrix} 1 & \phi_1(x^1) & \phi_2(x^1) & \ldots & \phi_k(x^1) \\ 1 & \phi_1(x^2) & \phi_2(x^2) & \ldots & \phi_k(x^2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \phi_1(x^N) & \phi_2(x^N) & \ldots & \phi_k(x^N) \end{pmatrix} \in \mathbb{R}^{N\times(k+1)}.$$

Observation: If you choose d feature functions as $\phi_i(x) = x_i$, for $i = 1, \ldots, d$, then the data matrix X is exactly the same as the one provided at the beginning of the exercise.

Exercise 2. (Linear regression and cross-validation)

A server is one of the main energy-consuming components of a data center. It has been found that the variables CPU denoted by $x_1 \in \mathbb{R}$, and the memory load denoted by $x_2 \in \mathbb{R}$, are two of the main contributing factors to the energy consumption of a server. You have made measurements of the CPU x_1^i, memory loads x_2^i, and energy consumption y^i, for $i = 1, 2, \ldots, 2000$ instances and aim to use linear regression to come up with a function that predicts a server's energy consumption. You randomly select 400 data samples for testing and the rest for training.

1. You have found that increasing CPU and memory load have a multiplicative effect on energy consumption. Hence, you define a new feature : $\phi(x_1, x_2) = x_1 x_2$. Write the equation for a linear predictor in terms of the features $x_1, x_2, \phi(x_1, x_2)$.

   Solution: The linear predictor is:

$$\hat{y} = b + w_1 x_1 + w_2 x_2 + w_3 x_1 x_2 = \begin{bmatrix} b & w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1 x_2 \end{bmatrix} = w^\top z,$$

   with $w, z \in \mathbb{R}^4$.

2. Write the regularized mean-square loss function for identifying the parameters of the model; use $\lambda \in \mathbb{R}$ for regularization.

   Solution: The regularized mean-square loss function is:

$$\begin{aligned} MSE(w) &= \frac{1}{1600} \sum_{i=1}^{1600} (y^i - w^\top z^i)^2 + \lambda w^\top w \\ &= \frac{1}{1600} (y_{train} - Z_{train} w)^\top (y_{train} - Z_{train} w) + \lambda w^\top w, \end{aligned}$$

   where $y_{train} \in \mathbb{R}^{1600}$ and $Z_{train} \in \mathbb{R}^{1600 \times 4}$ are obtained by stacking the y^i and the $(z^i)^\top$ respectively.

3. Derive the gradient of the loss function with respect to the linear regression parameters.

   Solution: The gradient of the mean-square loss function is:

$$\frac{\partial MSE}{\partial w} = \frac{2}{1600} Z_{train}^\top (Z_{train} w - y_{train}) + 2\lambda w.$$

4. Which is likely to decrease the training error: increasing or decreasing λ and why?

   Solution: To decrease the training error we need to decrease λ. We added the regularization parameter λ to reduce overfitting, so we use it to reduce the accuracy prediction on the

training data and thus we increase the training error. By setting λ equal to zero we minimize the training error.

5. Assume we choose the optimal λ using 5-fold cross-validation. Let $\hat{y}^i$ denote the prediction and y^i denote the actual server energy consumption for a given data point. How would you compute the mean validation error over the 5-folds?

   Solution: The mean validation error ϵ can be computed as the average of the error ϵ_f of each of the five folds

   $$\epsilon = \frac{1}{5}\sum_{f=1}^{5}\epsilon_f = \frac{1}{5}\sum_{f=1}^{5}\sum_{i\in I_f}\frac{1}{N_f}(\hat{y}^i - y^i)^2,$$

   where I_f gather the indices of the validation points in fold f and N_f is the cardinality of that fold.

6. Based on the result of 5-fold cross validation on the 1600 data points in the training set shown below, for which λ is the test error more likely to be similar to the validation error?

model	mean validation error	variance of error
λ_1	2.35	9.42
λ_2	1.30	4.16
λ_3	1.76	3.50

   Solution: It is λ_3, because it is the one with the lowest variance of error. A lower variance indeed indicates that this model is more robust against different data points being used as training points and hence more likely to perform similarly on some unseen data points.

Exercise 3. (Train vs. test datasets)

You want to predict whether or not a product fails based on historical data on the amount of the different materials used to make the product. You have a set of 1,000 data points, where each data point contains the amount of the 5 different materials ($x^i \in \mathbb{R}^d$) and the information on failure or non-failure of the product, $y^i \in \{0, 1\}$. Here 0 denotes no failure and 1 denotes failure.

1. Determine what type of learning problem you are dealing with: supervised or unsupervised learning? If supervised learning, is it a regression or classification problem?

   Solution: It is a supervised learning problem since we have data with labels of "failure, no failure". Furthermore, since the labels are finite, it is a classification problem.

2. You randomly split the data such that 800 data points are used as the training set and 200 are used as the test set. Why shouldn't we use all the 1,000 available data points to train the model?

   Solution: The test dataset should be separate from the training dataset because we want to ensure the classifier performs well on data it has never seen before.

3. After training your model, you observe that it performs well on the training data, but poorly on the test data. What is one possible explanation? What could you try to improve the performance on the test data?

   Solution: The classifier could be overfitting. Conceptually, it has "memorized" the training dataset instead of learning to generalize to the unseen test dataset. You could use regularization to fit a simpler model and thus avoid overfitting to improve the performance on the test data.