NYCU Introduction to Machine Learning, Homework 1

[112550198], [簡嫚萱]

Part. 1, Coding (60%):

(10%) Linear Regression Model - Closed-form Solution

1. (10%) Show the weights and intercepts of your linear model.

```
09-28 09:21:56.144 | INFO     | __main__:main:124 - LR_CF.weights=array([2.85501274, 1.01785863, 0.47202168, 0.19608925]), LR_CF.intercept=-33.695598
```

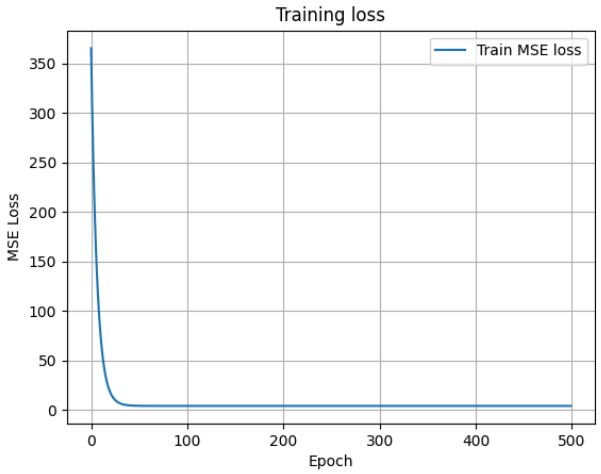
(40%) Linear Regression Model - Gradient Descent Solution

2. (10%)
   - Show the hyperparameters of your setting (e.g., learning rate, number of epochs, batch size, etc.).
   - Show the weights and intercepts of your linear model.

```
LR_GD.fit(train_x, train_y, epochs=500, learning_rate=0.04)
```

```
09-28 09:21:56.159 | INFO     | __main__:main:133 - LR_GD.weights=array([2.85501274, 1.01785863, 0.47202168, 0.19608925]), LR_GD.intercept=-33.695598
```

3. (10%) Plot the learning curve. (x-axis=epoch, y-axis=training loss)



4. (20%) Show your MSE.cf, MSE.gd, and error rate between your closed-form solution and the gradient descent solution.

```
Prediction difference: 0.00000000000030849812
mse_cf=4.29033059805880867543, mse_gd=4.29033059805880689908. Difference: 0.00000000000004140373%
```

(10%) Code Check and Verification

5. (10%) Lint the code and show the PyTest results.

```
(hw1) michelle@LAPTOP-E7SHGCCE:~/Machine_learning/hw1$ uv run flake8 main.py
(hw1) michelle@LAPTOP-E7SHGCCE:~/Machine_learning/hw1$
```

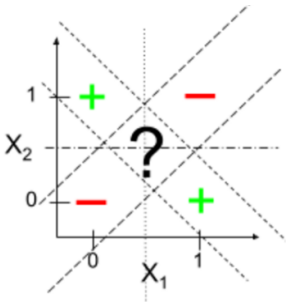
```
============================================================ test session starts ============================================================
platform linux -- Python 3.13.7, pytest-8.4.2, pluggy-1.6.0
rootdir: /home/michelle/Machine_learning/hw1
configfile: pyproject.toml
collected 2 items

test_main.py (100, 1)
2025-09-28 09:58:56.027 | INFO     | test_main:test_regression_cf:30 - model.weights=array([3.]), model.intercept=np.float64(4.000000000000004)
.(100, 1)
2025-09-28 09:58:56.028 | INFO     | main:fit:92 - EPOCH 0, loss=30606.0606, learning_rate=0.0001
2025-09-28 09:58:56.122 | INFO     | main:fit:92 - EPOCH 10000, loss=560.3453, learning_rate=0.0001
2025-09-28 09:58:56.217 | INFO     | main:fit:92 - EPOCH 20000, loss=10.2590, learning_rate=0.0001
2025-09-28 09:58:56.312 | INFO     | main:fit:92 - EPOCH 30000, loss=0.1878, learning_rate=0.0001
2025-09-28 09:58:56.407 | INFO     | main:fit:92 - EPOCH 40000, loss=0.0034, learning_rate=0.0001
2025-09-28 09:58:56.502 | INFO     | main:fit:92 - EPOCH 50000, loss=0.0001, learning_rate=0.0001
2025-09-28 09:58:56.597 | INFO     | main:fit:92 - EPOCH 60000, loss=0.0000, learning_rate=0.0001
2025-09-28 09:58:56.692 | INFO     | test_main:test_regression_gd:44 - model.weights=array([2.99999751]), model.intercept=np.float64(4.0000000000000036)
.

============================================================ 2 passed in 1.12s =============================================================
```

Part. 2, Questions (40%):

1. (10%) Linear models $y = w^\top x + b$ have limited fitting power.
   a. In one sentence, explain why a single linear model is limited.
   b. Give one concrete task that a single linear model cannot solve, and state why no single hyperplane/affine function solves it.

   a. A single linear model is limited by assuming a strictly linear input–output mapping, and lacks the capacity to approximate the nonlinear dependencies prevalent in real-world data.

   b. For instance, the XOR classification problem, which aims to predict the output of an XOR function given two binary inputs (as shown below), cannot be solved by a single linear model because the positive and negative examples are not linearly separable, meaning no single hyperplane can correctly divide the inputs into their corresponding classes.



2. (15%) Why do we add a regularization term in linear regression? What are the differences between L2 regularization (Ridge) and L1 regularization (Lasso)? Please explain in detail.

   Adding a regularization term in linear regression is to prevent overfitting and improve the stability of the solution by penalizing large weights. As for overfitting, if the number of features is large or features are highly correlated, the model may fit the training data very well but fail to generalize to new data. As for stability, when features are highly correlated(multicollinearity), the matrix $X^T X$ becomes nearly singular, and solving the normal equations can lead to very large or unstable weights. By adding a regularization term, we constrain the magnitude of the weights,

producing more stable solutions that generalize better to testing data. L2 regularization (Ridge) adds the squared values of the weights, which keeps all weights small but rarely exactly zero and reduces problems caused by correlated features. L1 regularization (Lasso) adds the absolute values of the weights, which can set some weights exactly to zero, effectively selecting important features and creating a simpler, sparse model. In short, Ridge shrinks all weights, while Lasso helps pick the most relevant features. Overall, Ridge aims to shrink all weights smoothly, while Lasso promotes sparsity and automatic selection of important features.

3. (15%)

- What is overfitting? Under what conditions can a model overfit? (List two) How can overfitting be alleviated? (List two)

Overfitting occurs when a model learns the training data too well, capturing not only the underlying patterns but also the noise, which leads to poor generalization of new data.

For example, a model is more likely to overfit when

1. It has too many parameters compared to the amount of training data
2. The dataset contains too much noise or highly similar features

Otherwise, overfitting can be alleviated by

1. Increasing the amount of training data
2. Adding regularization to constrain the model weights