NYCU Introduction to Machine Learning, Homework 1

[111550012], [鄭睿宏]

Part. 1, Coding (60%):

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

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

```
2024-10-07 18:52:30.272 | INFO     | __main__:main:87 - Closed-form weights: [2.8491883  1.0188675  0.48562739 0.1937254 ], intercept: -33.8223
```

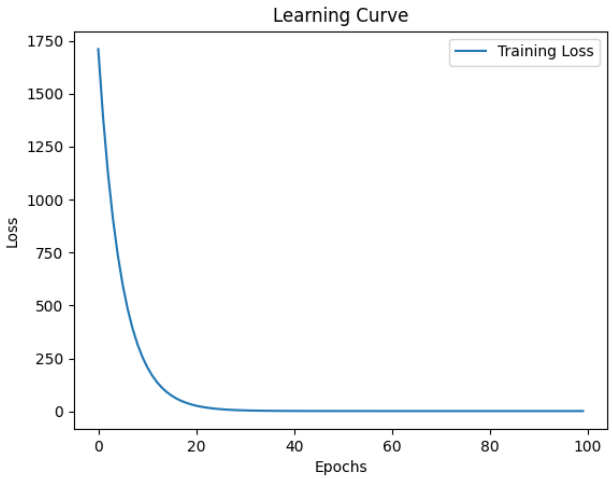
(40%) Linear Regression Model - Gradient Descent Solution

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

```
LR_GD.fit(train_x, train_y, learning_rate=0.1, epochs=100)
```

```
2024-10-07 18:52:33.729 | INFO     | __main__:main:93 - Gradient Descent weights: [2.8491883  1.0188675  0.48562739 0.1937254 ], intercept: -33.8223
```

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.

```
2024-10-07 18:52:33.732 | INFO     | __main__:main:110 - MSE (Closed-form): 4.1997, MSE (GD): 4.1997, Difference: 0.00%
```

(10%) Code Check and Verification

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

```
============================= test session starts =============================
platform win32 -- Python 3.11.9, pytest-8.3.3, pluggy-1.5.0
rootdir: C:\Users\Ray\Downloads\release
collected 2 items

test_main.py 2024-10-07 19:04:41.703 | INFO     | test_main:test_regression_cf:27 - model.weights=array([[3.]]), model.intercept=array([4.])
2024-10-07 19:04:41.703 | INFO     | main:fit:54 - Epoch 0, Loss: 15311.7162
2024-10-07 19:04:41.758 | INFO     | main:fit:54 - Epoch 10000, Loss: 2072.0082
2024-10-07 19:04:41.822 | INFO     | main:fit:54 - Epoch 20000, Loss: 280.3878
2024-10-07 19:04:41.880 | INFO     | main:fit:54 - Epoch 30000, Loss: 37.9426
2024-10-07 19:04:41.935 | INFO     | main:fit:54 - Epoch 40000, Loss: 5.1345
2024-10-07 19:04:41.995 | INFO     | main:fit:54 - Epoch 50000, Loss: 0.6948
2024-10-07 19:04:42.058 | INFO     | main:fit:54 - Epoch 60000, Loss: 0.0940
2024-10-07 19:04:42.126 | INFO     | test_main:test_regression_gd:39 - model.weights=array([2.99726525]), model.intercept=np.float64(3.9963467768899728)
.

============================== 2 passed in 1.01s ==============================
```

Part. 2, Questions (40%):

1. (10%) How does the presence of outliers affect the performance of a linear regression model? How should outliers be handled? List at least two methods.
    - Linear regression, being sensitive to extreme values, can be significantly impacted by outliers. These data points can introduce noise into the model, leading to increased variance and reduced predictive accuracy. By attempting to fit the outliers rather than the general trend in the data, the model's performance can be distorted.
    - The first method is detection, we can use boxplot to find the outlier and replace it with the nearest non-outlier value. This can help preserve the overall shape of the data.
    - The second method is RANSAC (Random Sample Consensus), which naturally ignores outliers by focusing only on the subset of points that fit the model well.

2. (15%) How do different values of learning rate (too large, too small…) affect the convergence of optimization? Please explain in detail.

- If the learning rate is excessively large, the optimization process may cause gradient explor ation, made the weights and interception goes to infinity. This can lead to oscillations or di vergence, where the model's parameters fluctuate wildly without settling at a stable solutio n. In extreme cases, the model may, cause the algorithm missed the minima loss function, and fail to converge entirely.
- If the learning rate is too small will result in slow convergence. The model takes only tiny steps toward the optimal solution, which can significantly increase the time it takes to reac h the minimum. In some cases, the model might get stuck in local minima, as the updates a re too small to escape such points. This can be especially problematic with large datasets o r complex models where faster convergence is needed.
- Finding the right learning rate means striking a compromise between stability and fast con vergence. By doing so, the model is able to move closer to the ideal solution without overs hooting or being trapped in local minima. Throughout the training phase, adaptive learning rate strategies like learning rate schedules or optimizers like Adam can support the preserv ation of this equilibrium.

3. (15%)

- What is the prior, likelihood, and posterior in Bayesian linear regression. [Explain the concept in detail rather than writing out the mathematical formula.]
- What is the difference between Maximum Likelihood Estimation (MLE) and Maximum A Posteriori Estimation (MAP)? (Analyze the assumptions and the results.)

Prior Distribution: The initial belief about the parameters before observing the data,such as the weights in linear regression. This belief could be based on previous experience, knowledge, or assumptions about the problem.

Likelihood: The probability density values of the residuals given a specific set of parameters. It tells us how likely the data is under different assumptions about the relationship. If the parameters result in predictions that closely match the actual data, the likelihood will be high.

Posterior Distribution: The updated belief about the parameters after observing the data. It combines the prior distribution and the likelihood to provide a more informed estimate of the parameters. The posterior helps us revise our belief about the parameters based on the new data and gives us a more accurate estimate.

- MLE (Maximum Likelihood Estimation):
  - Assumption: MLE looks for values (weight) that maximize the likelihood of the data, assuming that the hyperparameters have fixed values.
  - Result: MLE provides a point estimate of the parameters. It is sensitive to outliers and may not be robust to small amount of data.

$$\theta_{MLE} = argmax_\theta \sum_i \ln p(y_i \mid x_i, m, \theta)$$

- MAP (Maximum A Posteriori Estimation):
  - Assumption: MAP incorporates both the likelihood and prior distribution for the parameters, reflecting our prior beliefs about the parameters. It seeks to find the values that maximize the posterior distribution.
  - Result: Like MLE, MAP provides a point estimate of the parameters. However, it is less sensitive to outliers and can be more robust to limited sample sizes due to the influence of the prior.

$$\theta_{MAP} = argmax_\theta \sum_i \ln p(y_i \mid x_i, m, \theta) + \ln p(\theta \mid m)$$

Fundamentally, MLE is a frequentist method that only rely on likelihood, whereas MAP is a Bayesian method that combines the likelihood with prior knowledge in the hopes that θ can approach zero. Typically, we consider $\boldsymbol{p(\theta|m)}$ to be a variance limited method whose mean equals zero.