NYCU Introduction to Machine Learning, Homework 1

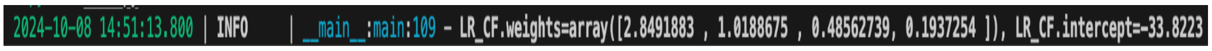
[111550086], [陳冠智]

The screenshot and the figures we provided below are just examples. **The results below are not guaranteed to be correct.** Please make sure your answers are clear and readable, or no points will be given.
Please also remember to convert it to a pdf file before submission.
You should use English to answer the questions (-5pt if not report in English).
After reading this paragraph, you should delete this paragraph.

Part. 1, Coding (60%):

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

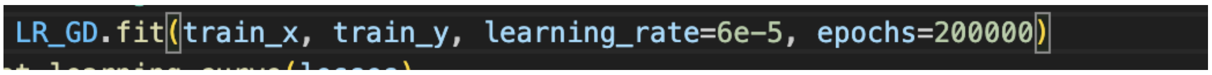
1. (10%) Show the weights and intercepts of your linear model.



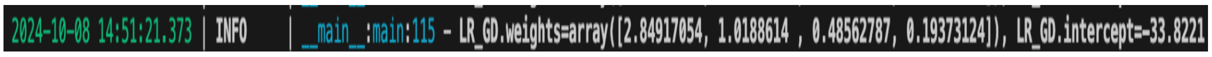
```
2024-10-08 14:51:13.800 | INFO     | __main__:main:109 - LR_CF.weights=array([2.8491883 , 1.0188675 , 0.48562739, 0.1937254 ]), LR_CF.intercept=-33.8223
```

(40%) Linear Regression Model - Gradient Descent Solution

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

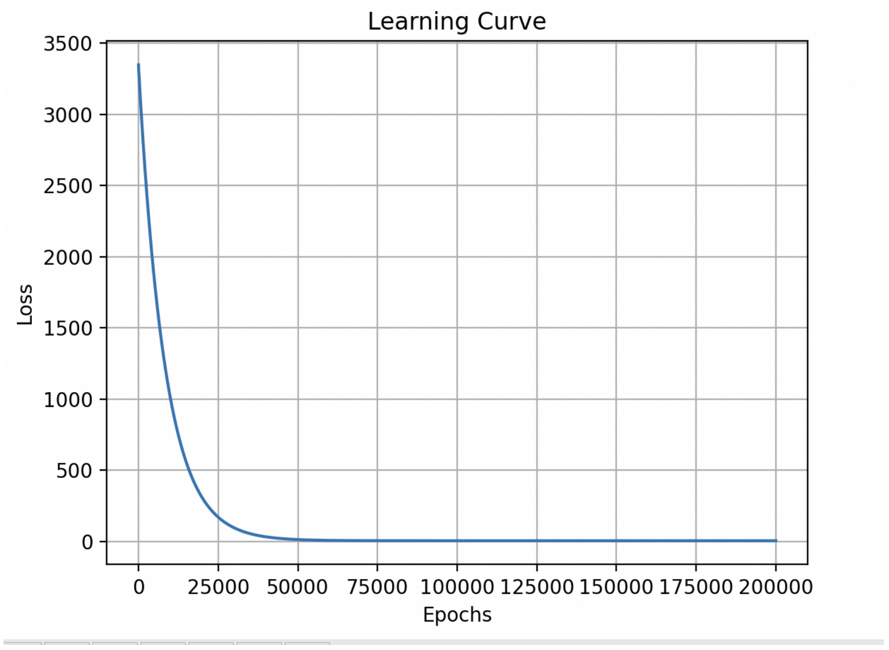


```
LR_GD.fit(train_x, train_y, learning_rate=6e-5, epochs=200000)
```

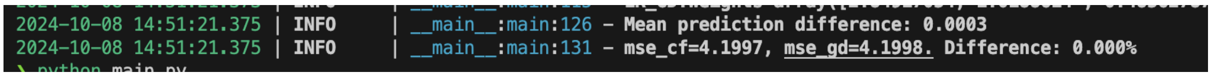


```
2024-10-08 14:51:21.373 | INFO     | __main__:main:115 - LR_GD.weights=array([2.84917054, 1.0188614 , 0.48562787, 0.19373124]), LR_GD.intercept=-33.8221
```

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



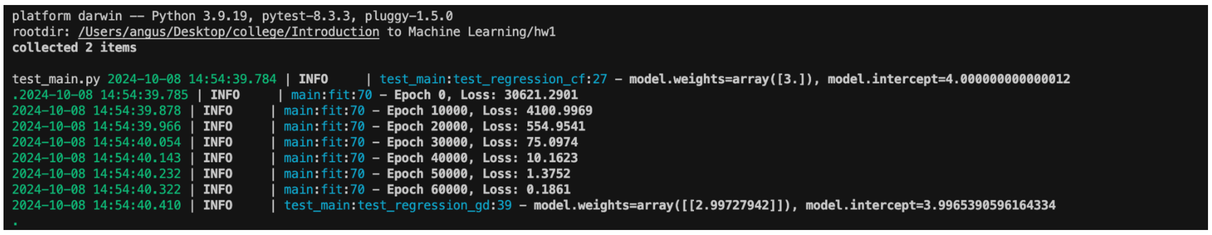
4. (20%) Show your MSE.cf, MSE.gd, and error rate between your closed-form solution and the gradient descent solution.



```
2024-10-08 14:51:21.375 | INFO     | __main__:main:126 - Mean prediction difference: 0.0003
2024-10-08 14:51:21.375 | INFO     | __main__:main:131 - mse_cf=4.1997, mse_gd=4.1998. Difference: 0.000%
```

(10%) Code Check and Verification

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



```
platform darwin -- Python 3.9.19, pytest-8.3.3, pluggy-1.5.0
rootdir: /Users/angus/Desktop/college/Introduction to Machine Learning/hw1
collected 2 items

test_main.py 2024-10-08 14:54:39.784 | INFO     | test_main:test_regression_cf:27 - model.weights=array([3.]), model.intercept=4.000000000000012
.2024-10-08 14:54:39.785 | INFO     | main:fit:70 - Epoch 0, Loss: 30621.2901
2024-10-08 14:54:39.878 | INFO     | main:fit:70 - Epoch 10000, Loss: 4100.9969
2024-10-08 14:54:39.966 | INFO     | main:fit:70 - Epoch 20000, Loss: 554.9541
2024-10-08 14:54:40.054 | INFO     | main:fit:70 - Epoch 30000, Loss: 75.0974
2024-10-08 14:54:40.143 | INFO     | main:fit:70 - Epoch 40000, Loss: 10.1623
2024-10-08 14:54:40.232 | INFO     | main:fit:70 - Epoch 50000, Loss: 1.3752
2024-10-08 14:54:40.322 | INFO     | main:fit:70 - Epoch 60000, Loss: 0.1861
2024-10-08 14:54:40.410 | INFO     | test_main:test_regression_gd:39 - model.weights=array([[2.99727942]]), model.intercept=3.9965390596164334
.
```

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.

1. Weighting Schemes:

Weighted Least Squares: Assign weights to data points based on their variance, giving less influence to outliers during the regression fitting process.

Moreover, I think it's important to mention that I normalized the training data before performing gradient descent. I noticed that the "Previous Scores" feature had values much larger than the other features, which could make gradient descent unstable and significantly increase the time it takes to converge to the optimal solution. Therefore, I normalized each feature

2. Data Transformation and Cleaning:

Winsorizing: Limiting extreme values in the data to reduce the effect of possibly spurious outliers.

Log Transformation: Applying a logarithmic transformation to compress the range of the data, which can mitigate the influence of outliers.

3. Robust Regression Methods:

Use of Robust Estimators: Techniques like Least Absolute Deviations (LAD) regression minimize the sum of absolute residuals rather than squared residuals, reducing the influence of outliers.

RANSAC (Random Sample Consensus): An iterative method that fits the model to random subsets of the data, isolating inliers from outliers and fitting the model primarily to the inliers.

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

1. Learning Rate Too Large:

Effect: When the learning rate is too large, the model takes very large steps during each update. Instead of gradually approaching the optimal solution, it overshoots, causing the loss function to oscillate or even diverge (move further away from the optimum).

Consequence:

1. Oscillations: The model parameters might oscillate around the optimal point, making it difficult to converge.
2. Divergence: The parameters might be pushed further away from the minimum, leading to an increase in the loss function over time.
3. Unstable Training: Large learning rates can result in unstable or erratic updates, making the training process unpredictable.

Visualization: The loss function might never decrease consistently, bouncing between high values across epochs.

2. Learning Rate Too Small:

Effect: When the learning rate is too small, the model takes very tiny steps toward the optimal solution. While this avoids the risk of overshooting, it can significantly slow down the convergence process.

Consequence:

1. Slow Convergence: The model may take many iterations to make meaningful progress, leading to extremely long training times.
2. Stuck in Local Minima: The model might get stuck in a local minimum or a saddle point because the steps are too small to escape these regions.
3. Inefficiency: Even though it may eventually converge, the small learning rate makes the process inefficient, requiring a large number of epochs to reach a reasonable solution.

Visualization: The loss function decreases very slowly over many iterations, sometimes appearing almost flat.

3. Optimal Learning Rate:

Effect: With an appropriately chosen learning rate, the model takes moderately sized steps that allow it to gradually converge to the optimal solution. The updates are large enough to make progress, but small enough to avoid overshooting.

Consequence:

1. Efficient Convergence: The model converges at a reasonable speed without oscillating or diverging.
2. Stable Training: The loss function steadily decreases with minimal noise or fluctuations.

Visualization: The loss function decreases smoothly over time, reflecting efficient and stable training.

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.)

1.Prior, Likelihood, and Posterior in Bayesian Linear Regression:

1) Prior: The prior represents our initial beliefs about the regression parameters before observing any data. It's a probability distribution that reflects what we think the parameters might be based on prior knowledge or assumptions. For example, we might assume that the coefficients are likely to be around zero, indicating no strong effect.
2) Likelihood: The likelihood quantifies how probable the observed data is given specific parameter values. It represents the relationship between the parameters and the data under the statistical model, typically assuming that the residuals (errors) are normally distributed in linear regression.
3) Posterior: The posterior combines the prior and the likelihood to update our beliefs about the parameters after observing the data. It represents the updated probability distribution of the parameters, reflecting both prior beliefs and the new information from the data. This update is performed using Bayes' theorem: Posterior $\propto$ Likelihood $\times$ Prior.

2.Difference Between Maximum Likelihood Estimation (MLE) and Maximum A Posteriori Estimation (MAP):

2.1 Maximum Likelihood Estimation (MLE):

1. Assumptions: Relies solely on the observed data and the likelihood function without incorporating any prior beliefs.
2. Approach: Estimates parameters by finding the values that maximize the likelihood function, essentially identifying the parameter values that make the observed data most probable under the model.
3. Results: Provides point estimates based only on the data, which can lead to overfitting, especially when the dataset is small.

2.2 Maximum A Posteriori Estimation (MAP):

1. Assumptions: Incorporates both the likelihood function and a prior distribution over the parameters, reflecting prior knowledge or beliefs.
2. Approach: Estimates parameters by finding the values that maximize the posterior distribution, balancing the influence of the data and the prior beliefs.
3. Results: Produces estimates that are more robust to overfitting by considering prior information, which is particularly beneficial when data is scarce.

3.Conclusion:

1. MLE is suitable when there is little to no prior knowledge about the parameters or when we want the data to speak entirely for itself.
2. MAP is advantageous when prior information is available and can enhance parameter estimation, especially in situations with limited data or when regularization is desired.