Principle of Artificial Intelligence HW1:
0811562 何祁恩 0810761 羅雯瀠 (No information for the third person … 3 email without reply)

Part 1: Linear Regression + SGD

1. Model Parameter

$$\begin{cases} iteration\ T = 1000 \\ learning\ rate\ \eta = 0.01 \\ loss\ function\ Loss = Square\ Loss \end{cases}$$

2. Gradient Calculation Detail

Let $\hat{y} = w \cdot x + b$, $Loss = (y - \hat{y})^2 = (y - [w \cdot x + b])^2$

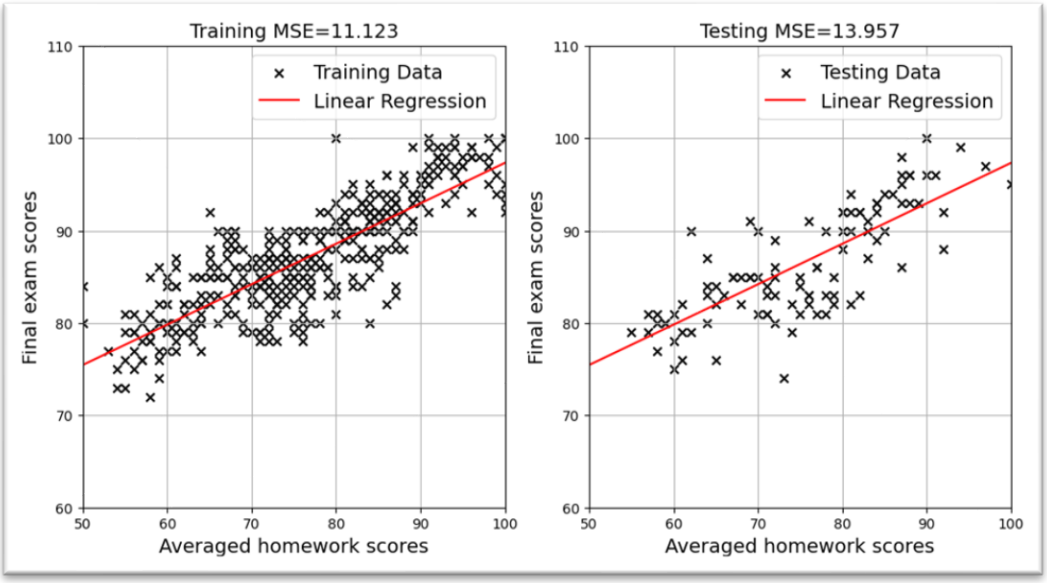
- Partial Derivative on w & b:

$$\begin{cases} \dfrac{\partial Loss}{\partial w} = 2 \cdot (y - \hat{y}) \cdot (-x) = 2 \cdot x \cdot (\hat{y} - y) \\ \dfrac{\partial Loss}{\partial b} = 2 \cdot (y - \hat{y}) \cdot (-1) = 2 \cdot (\hat{y} - y) \end{cases}$$

- Update w & b through:

$$\begin{cases} w = w - \eta \dfrac{\partial Loss}{\partial w} = w - \eta \cdot [2 \cdot x \cdot (\hat{y} - y)] \\ b = b - \eta \dfrac{\partial Loss}{\partial b} = b - \eta \cdot [2 \cdot (\hat{y} - y)] \end{cases}$$

3. Result:



As you can see from the above figure, the training MSE (11.123) is smaller than the testing data MSE (13.957). The linear regression line is well-fitted between the averaged homework scores and final exam scores.

Part 2: Logistic Regression + SGD

1. Model Parameter

$$\begin{cases} iteration\ T = 1000 \\ learning\ rate\ \eta = 0.75 \\ loss\ function\ Loss = Logistic\ loss \end{cases}$$

2. Gradient Calculation Detail

Let $z(w_i) = \sum_{i=0}^{2} w_i x_i$ be the inner product between the data & weight.

, where $\begin{cases} x_0 = 1 \\ x_1 = Normalized\left(Averaged\ Homework\ Scores\right) \\ x_2 = Normalized\left(Final\ Exam\ Scores\right) \end{cases}$

- Pass inner product of weight & data to the sigmoid function:

$$p(z) = \sigma(z) = \frac{1}{1+e^{-z}}$$

- By the definition of the logistic loss:

$$L(p) = \begin{cases} -\log(1-p), y = 0 \\ -\log(p), y = 1 \end{cases} = -y \cdot \log p - (1-y) \cdot \log(1-p)$$

, where y is the true labeling data.

- By the chain rule of the partial derivative:

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial p} \cdot \frac{\partial p}{\partial z} \cdot \frac{\partial z}{\partial w_i}, \forall i = 0,1,2$$

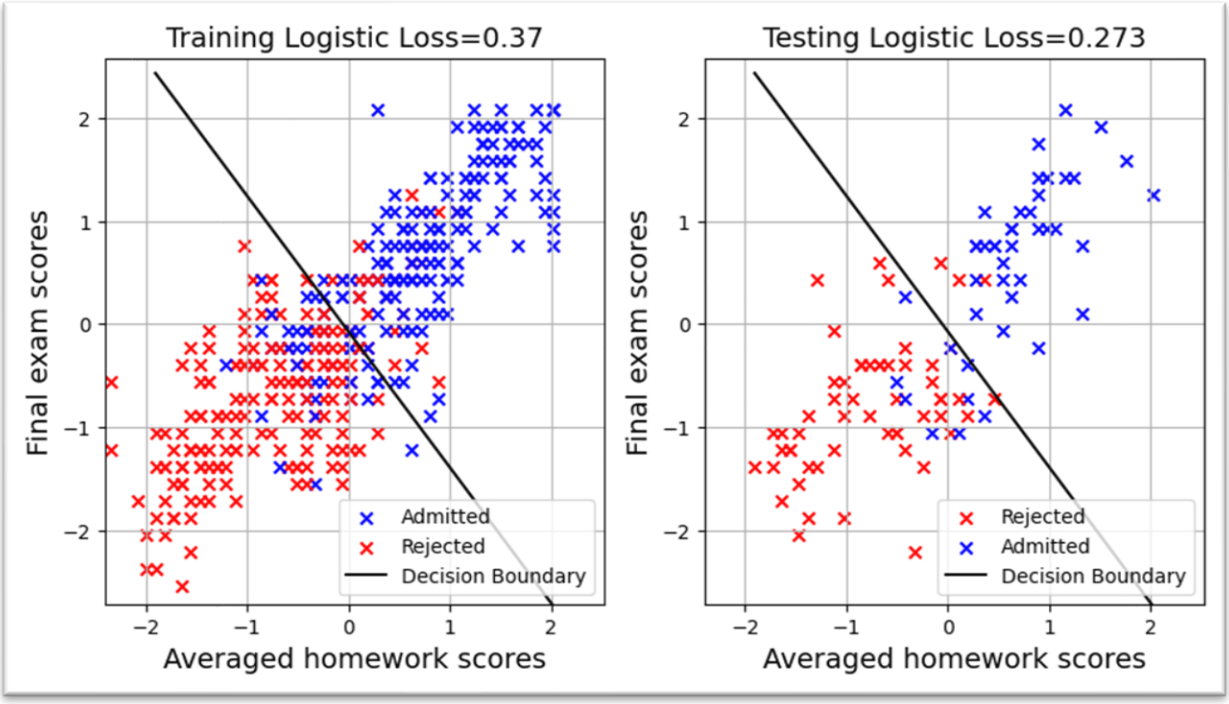
$$\begin{cases} \frac{\partial L}{\partial p} = \frac{\partial}{\partial p}\left[-y \cdot \log p - (1-y) \cdot \log(1-p)\right] = \left[\left(\frac{-y}{p}\right) + \left(\frac{1-y}{1-p}\right)\right] \\ \frac{\partial p}{\partial z} = \frac{\partial}{\partial z}\left[\frac{1}{1+e^{-z}}\right] = \frac{e^{-z}}{\left(1+e^{-z}\right)^2} = \left[\frac{1}{\left(1+e^{-z}\right)} \cdot \frac{e^{-z}}{\left(1+e^{-z}\right)}\right] = p \cdot (1-p) \\ \frac{\partial z}{\partial w_i} = \frac{\partial}{\partial w_i}\left[\sum_{i=0}^{2} w_i x_i\right] = x_i \end{cases}$$

$$\text{Therefore, } \frac{\partial L}{\partial w_i} = \left[\left(\frac{-y}{p}\right) + \left(\frac{1-y}{1-p}\right)\right] \cdot \left[p \cdot (1-p)\right] \cdot x_i = (p-y) \cdot x_i, \forall i = 0,1,2$$

- Update w through:

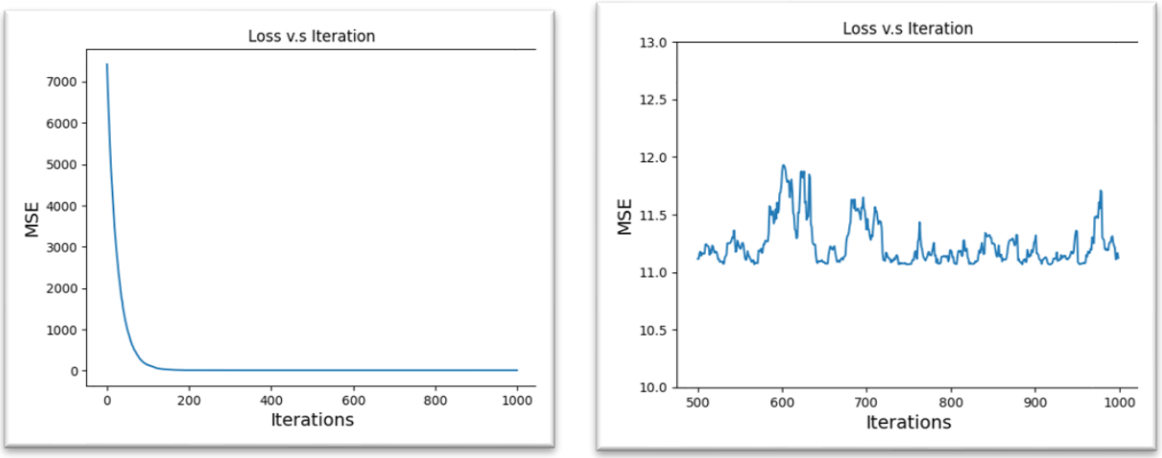
$$w_i = w_i - \eta \frac{\partial L}{\partial w_i} = w_i - \eta \cdot \left[(p-y) \cdot x_i\right], \forall i = 0,1,2$$

3. Result:

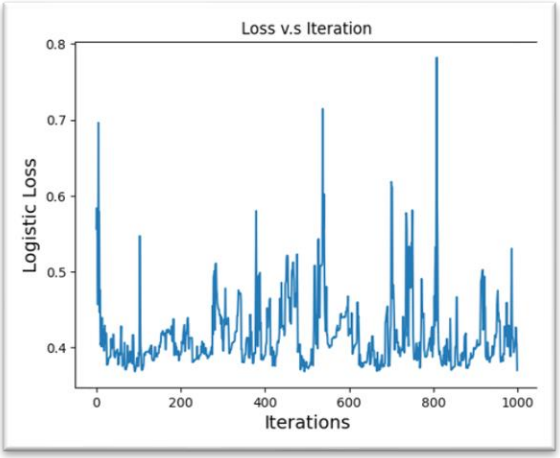


In the training data, some of the admitted individuals had lower average and final scores compared to those who were rejected. It is possible that their involvement in a research project significantly boosted their scores, leading to their successful admission to NYCU's master's program. Additionally, there were some candidates who performed exceptionally well in both assessments but were not admitted to NYCU's master's program. This could be due to their grades being inflated, and they might have struggled to answer questions during the interview process. Consequently, it is challenging to find a clear boundary for distinguishing between admissions and rejections, leading to a higher loss in the training data. In contrast, in the testing data, as such trends were less observed, the loss was relatively lower.

Part 3: Hyperparameter selection



From the above loss versus iterations graph, we can see that the MSE loss actually decreases in each iteration. However, if we monitor the last 500 iterations, we can observe that the loss does not converge due to the learning rate being too high at that time. Same trend can be observed in logistic regression in the graph in the next page. So, let’s experiment with the parameters to fine-tune the model.



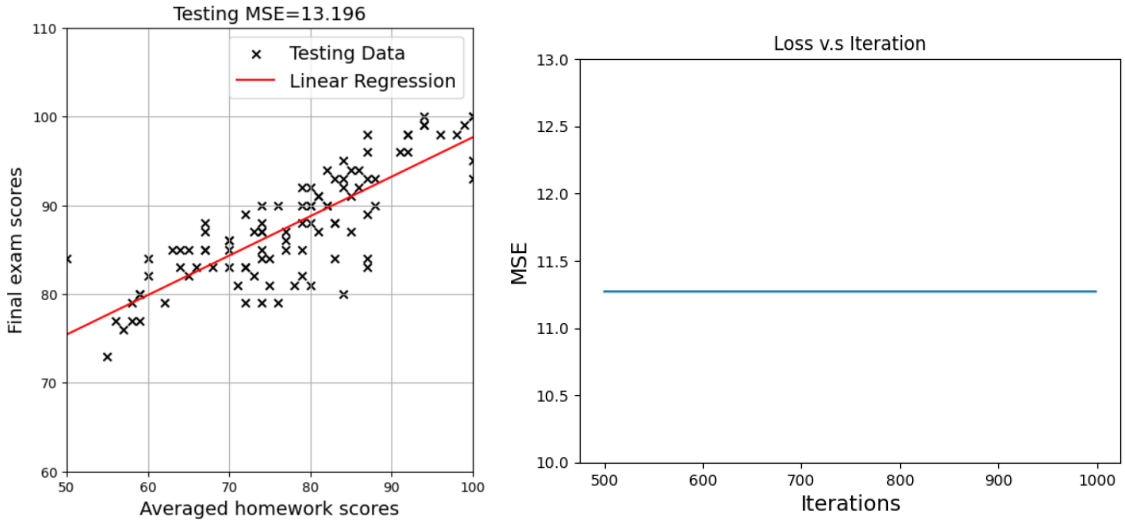
Part 3-1: Linear Regression + SGD Hyperparameter selection

1. Model Parameter

$$\begin{cases} iteration\ T = 1000 \\ learning\ rate\ \eta = [0.01, 0.005, 0.0001, 0.00001] \\ batch\ size\ = 5 \\ loss\ function\ Loss = Square\ Loss \end{cases}$$

To address the issue of the learning rate being too large for the model to converge, I have implemented a learning rate schedule. This schedule reduces the learning rate after a certain number of iterations. Additionally, to mitigate the influence of outliers during the Stochastic Gradient Descent process, I have opted for mini-batches with a batch size of 5. This approach reduces the impact of outliers on the overall training process. Moreover, to ensure that the data is not biased towards any particular distribution, I have shuffled the data beforehand. The iteration count, T, remains unchanged, allowing the model to train efficiently while achieving higher accuracy.

2. Result



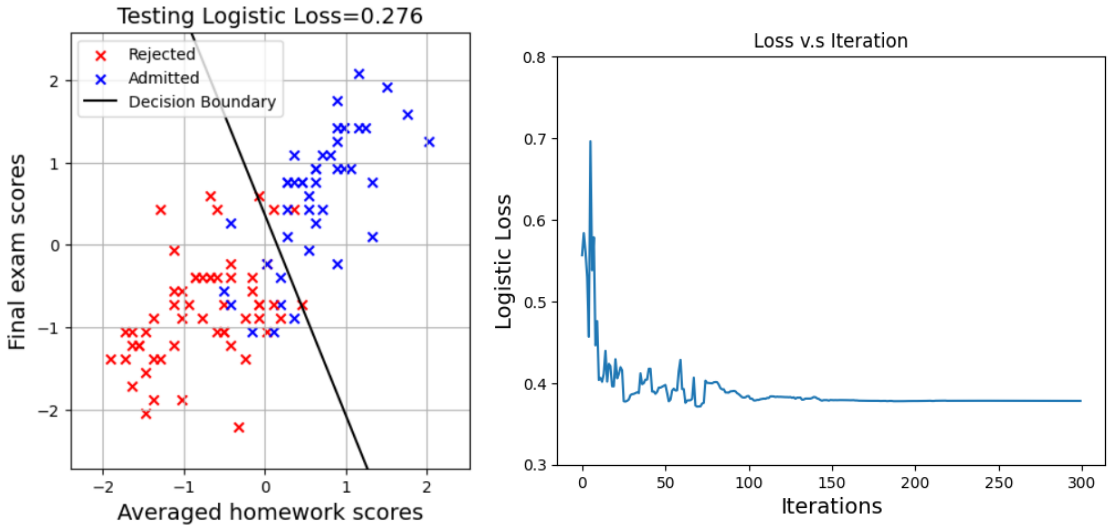
As you can see, the testing MSE is less than the previous model and the loss converge to approximate 11.3 before the 500 iterations. That is, I can stop the training process before 500 iterations and also get a high accuracy model.

Part 3-2: Logistic Regression + SGD Hyperparameter selection

1. Model Parameter

$$\begin{cases} iteration\ T = 300 \\ learning\ rate\ \eta = [0.75, 0.075, 0.0075, 0.00075] \\ loss\ function\ Loss = Logistic\ Loss \end{cases}$$

To address the issue of fluctuating logistic loss caused by a large learning rate, a modified learning rate schedule can be employed to achieve quicker convergence without sacrificing accuracy. Additionally, the existing model from Part 2 demonstrates satisfactory accuracy. Consequently, it is imperative to concentrate on refining the convergence rate without compromising the model's accuracy. Adjusting the learning rate schedule can be an effective strategy to achieve this goal.



It's great to see that by adjusting the learning rate schedule, the testing loss remains stable and the training iteration is reduced by 70%, leading to a substantial enhancement in training efficiency without compromising accuracy. This improvement highlights the importance of carefully managing the learning rate schedule to achieve a balance between convergence speed and model accuracy. Adjusting this schedule is a crucial step in optimizing the training process.

Summary

This lab focused on the implementation of a linear regression & logistic regression model using stochastic gradient descent for regression & binary classification. We processed the dataset, applied feature normalization, and incorporated a learning rate schedule to enhance the convergence speed of the model. By introducing a mini-batch approach and adjusting the learning rate based on predefined thresholds, we effectively improved the training efficiency without compromising the accuracy of the model. The results highlighted the significance of fine-tuning the learning rate schedule to achieve a balance between convergence speed and accuracy in training machine learning models.