NYCU Intro. to ML
HW1 Report

110550093, 蔡師睿

Part I. Coding

Linear Regression Model - Closed-form Solution

1. Show the weights and intercepts of your linear model.

```
Closed-form Solution
Weights: [2.85817945 1.01815987 0.48198413 0.1923993 ], Intercept: -33.78832665744901
```

Linear Regression Model - Gradient Descent Solution

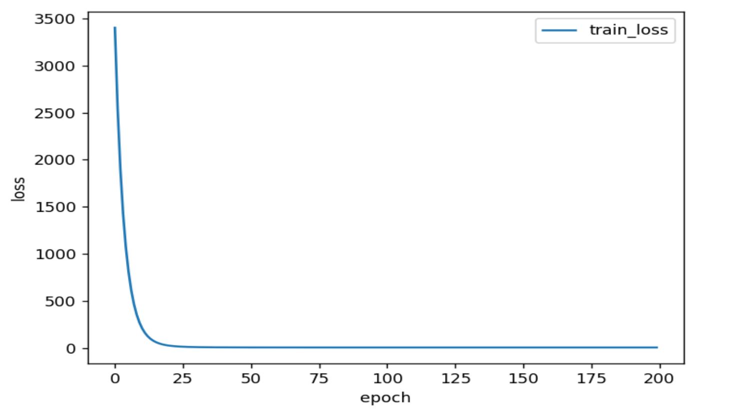
2. Show the learning rate and epoch you choose.

```
LR.gradient_descent_fit(train_x, train_y, lr=0.9, epochs=200)
```

3. Show the weights and intercepts of your linear model.

```
Gradient Descent Solution
Weights: [2.85805228 1.01813389 0.48181296 0.19228043], Intercept: -33.78411030203452
```

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



5. Show you error rate between your closed-form solution and the gradient descent solution.

```
Error Rate: -0.0%
```

Part II. Questions

1. How does the value of learning rate impact the training process in gradient descent?

The value of learning rate can determine the size of the steps the model takes. If a high learning rate is set, the steps will become large, and the model will converge quickly as well. However, it may sometimes result in divergence or oscillation. By contrast, if a low learning rate is set, which means the steps are small, the model will converge slowly, and it needs much more training times to reach a better model performance.

2. There are some cases where gradient descent may fail to converge. Please avoid at least two scenarios and explain in detail.

1. Saddle Point : There exists saddle points in higher dimensional space, so when applying gradient descent, it will stuck in the local minimum sometimes, and the model will think that the local minimum is the best solution. To solve the problem, we can try initializing parameters randomly, or there is a more effective method, which is to apply momentum in gradient descent to escape saddle points .

2. Poor Feature Scaling : Gradient descent in linear regression will take longer to converge for larger ranges features or even fail to converge, because the features in the dataset have different ranges significantly. Thus, to avoid the scenario, it's necessary to do normalization or standardization to the data.

3. Is mean square error (MSE) the optional selection when modeling a simple linear regression model? Describe why MSE is effective for resolving most linear problems and list scenarios where MSE may be inappropriate for data modeling, proposing alternative loss functions suitable for linear regression modeling in those cases.

Yes, MSE is the optional selection when modeling a simple linear regression model. Because MSE is the averaged squared difference between predicted values and ground truth, it is intuitive and easy to work with. Moreover, MSE is differentiable, which make it suitable to optimize in gradient descent.

However, there still some cases that MSE may be inappropriate. Some examples are as follows :
1. When extreme outliers exist in the data, MSE will give more weight to the outlier data points due to its sensitivity to outliers. Instead, we can use mean absolute error (MAE) or Huber loss as alternative loss functions.
2. When encounter the classification problems, MSE may be not suitable. Instead, cross entropy loss may be more suitable.

4. In the lecture, we learned that there is a regularization method for linear regression models to boost the model's performance.

$$E_D(w) + \lambda E_W(w)$$

4.1 Will the use of the regularization term always enhance the model's performance?

Not necessarily always better or worse.

4.2 We know that λ is a parameter that should be carefully tuned.

4.2.1 Discuss how the model's performance may be affected when λ is set too small.

If λ is set to be too small, the regularization term impacts on the loss function becomes weak, and it won't prevent overfitting. Hence, with the small value of λ, the performance of the model may still lead to overfit, and the weights may still become extreme large.

4.2.2 Discuss how the model's performance may be affected when λ is set too large.

If λ is set to be too large, the model will place excessive emphasis on regularizing the weights rather than minimizing the loss between the predicted and ground truth values. This can lead to the model prioritizing the penalty for large weights while neglecting its performance.