Here is the link of the code:

For this week’s honor, I tried to implement Single Linear Regressions with PyTorch. The main idea was to use one independent numerical variable to predict the value of the other numerical variables.

The first thing I did was to randomly find a dataset with numerical data that I could use. I decided to use the Seattle Housing dataset and the review scores rating to predict the price of the house. After that, I divided the dataset into training and testing datasets. Since it is a single linear regression, I did not use validation data to improve the accuracy of the trained model. I also converted the data type from series to numpy so that I can convert it to tensor later. So, all the preparations were done for neural network training.

Second, I used PyTorch to build a single linear regression model. Since the implementation of neural networks is complex, I chose to inherit the neural network class provided by torch libraries by adding input size and output size as additional parameters. I also implemented the forward function for forward propagation and prediction purposes. As for other functions, like backward propagation, I decided to use the function provided by the torch neural network class. As for the loss function, I chose to use mean square error, which guarantees to be positive and emphasizes the difference between actual and predicted value. As for the learning rate, I set it to 0.0001 so that the result will be more accurate. However, compared with a larger learning rate, the runtime may be longer. As for the optimizer, I used SGD to find the point where the derivative of loss is zero.

Finally, I started the training process. To begin with, I set the input size and output size to 1 to ensure one input will give one output. Then, I wrote a for loop to repeat the training process for 100 times to find the optimal parameters for the trained model. In each loop, I convert the numpy data to tensor so that PyTorch can use them and get predicted data from the independent variable. In this case, I put “review scores rating” to the model and get the predicted price. Then, use the loss function to find the loss and use the optimizer to update the parameters. Finally, after training 100 times, we get a linear model to predict the price of the house from the review scores rating of the houses.

However, based on the model I got, the mean square error is very large. There are several reasons that may cause this error. First, the data I used to train the model is not linear, which is true for the dataset I used. There are other variables that will contribute to the price of the houses. I can use multiple linear regression later to see if it can solve the problem. Second, since I am not very familiar with PyTorch, there may be some problems with my codes, which I should fix later.