Claude Shyaka
ID#: 801326243

Homework 0: Linear Regression with Gradient Descent

Link to the GitHub repo: https://github.com/claudeshyaka/ml

Problem 1.

In this exercise, a linear regression model with gradient descent algorithm with one explanatory variable as input was developed and the following results were obtained.

For each explanatory variable, the linear regression model was run with the following parameters:

- An alpha value of 0.04,
- Number of iterations of 2000,
- theta vector initialized to (0., 0.).

Explanatory variable X1,

- Final theta vector: **[5.928, -2.038],** rounded to three decimal places.
- Final loss value of approximately **0.985**, rounded to three decimal places.

Figure 1 shows the final regression model for the explanatory variable X1.

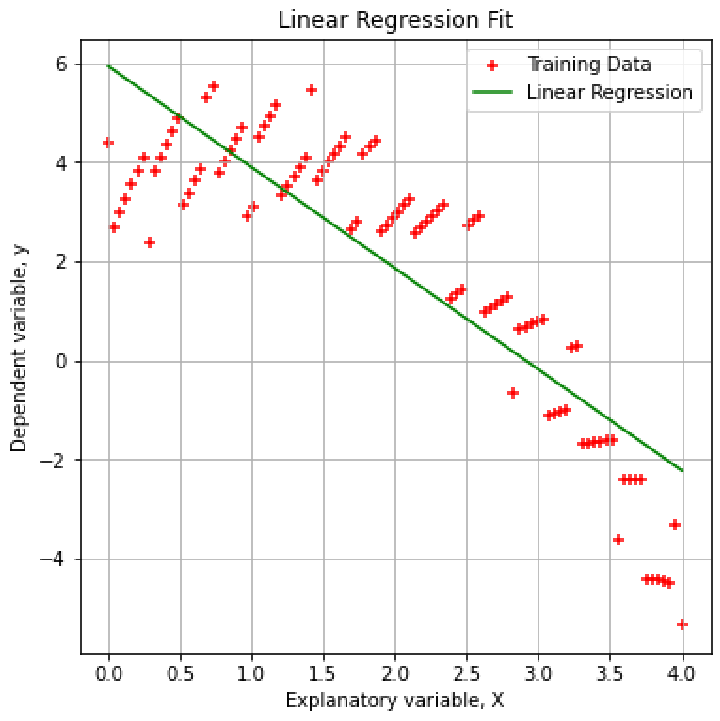

Figure 1: Final regression model for X1 and y

Figure 2 shows the loss over the number of iterations for X1.

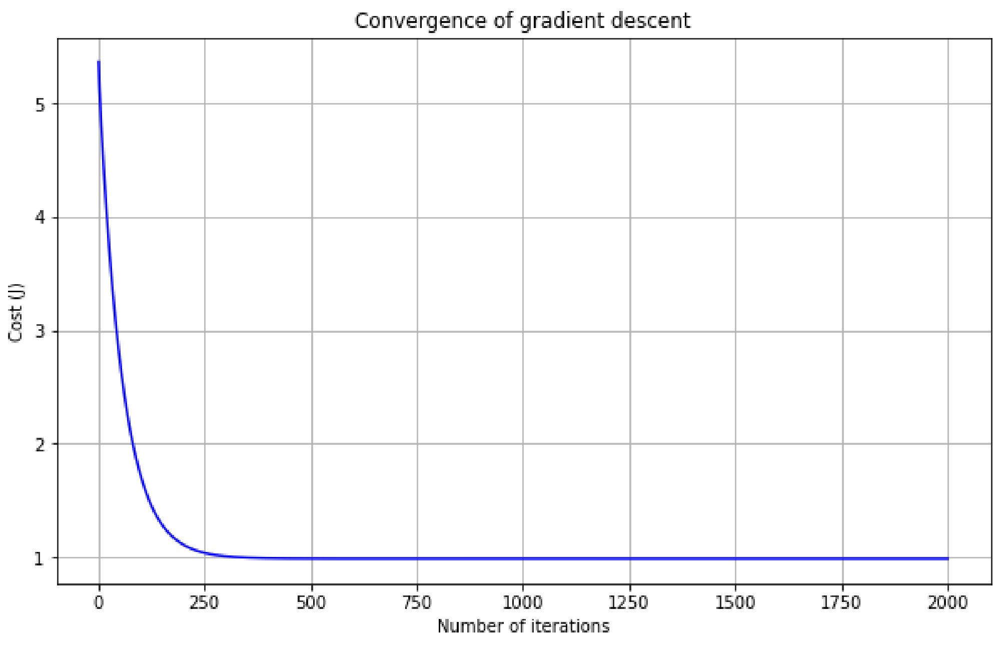

Figure 2: Loss (J) over the number of iterations for X1.

Explanatory variable X2,

- Final theta value: **[0.736, 0.558],** rounded to three decimal places.
- Final loss value of approximately **3.599,** rounded to three decimal places.

Figure 3 shows the final regression model for the explanatory variable X2.

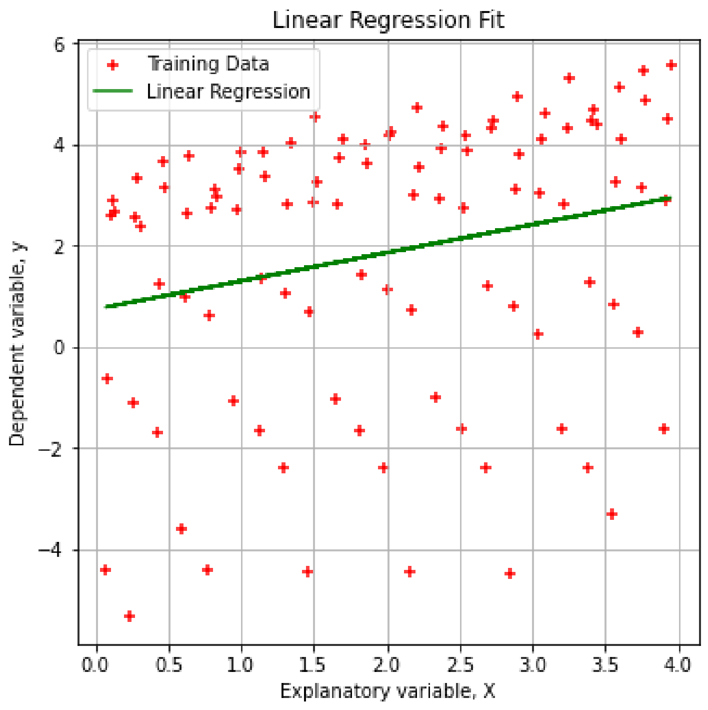

Figure 3: Final regression model for X2 and y

Figure 4 shows the loss over the number of iterations for X2

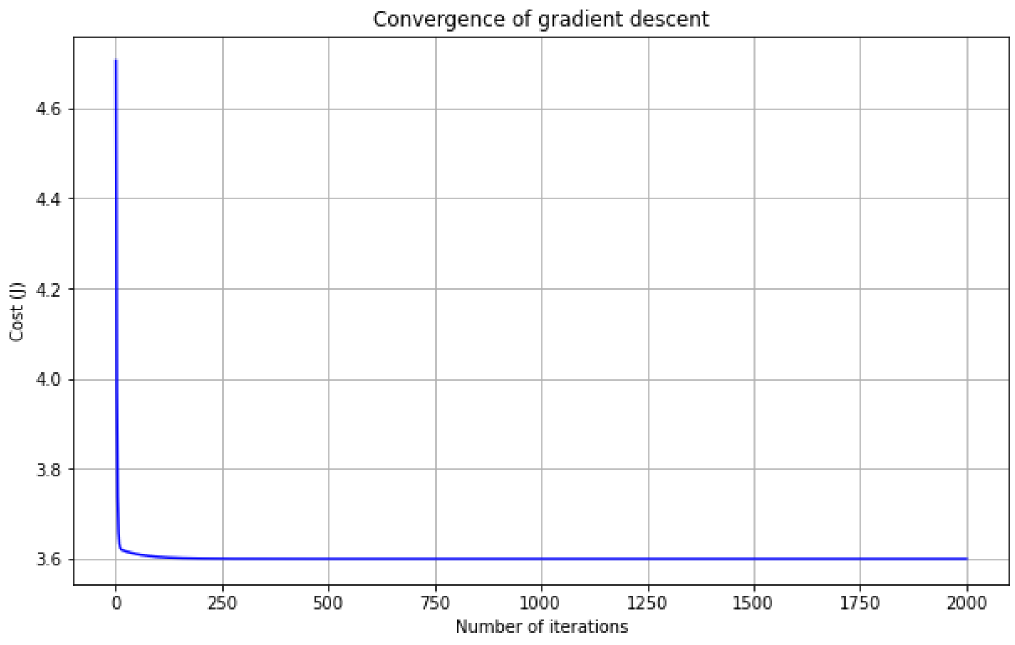

Figure 4: Loss (J) over the number of iterations for X2.

Explanatory variable X3,

- Final theta value: **[2.871, -0.520],** rounded to three decimal places.
- Final loss value of approximately **3.629,** rounded to three decimal places.

Figure 5 shows the final regression model for the explanatory variable X3.

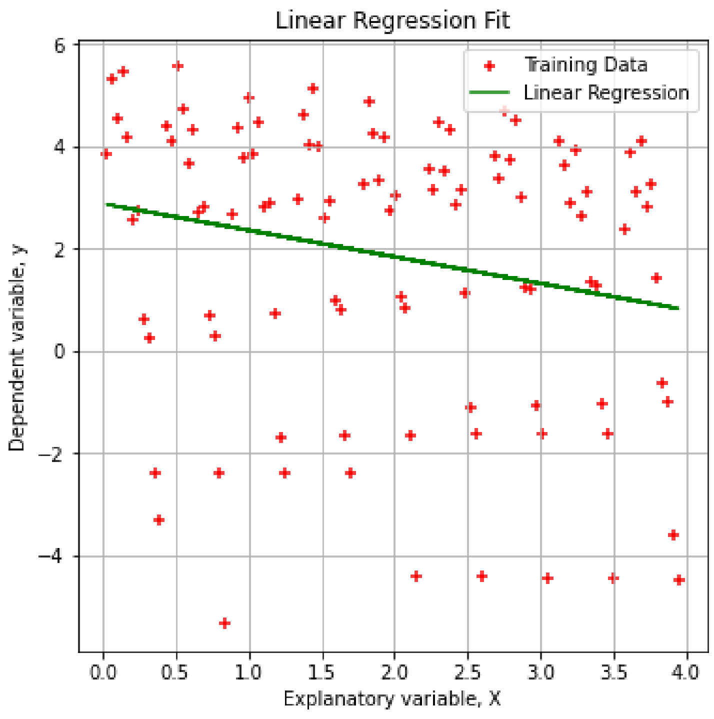

Figure 5: Final regression model for X3 and y

Figure 6 shows the loss over the number of iterations for X3.

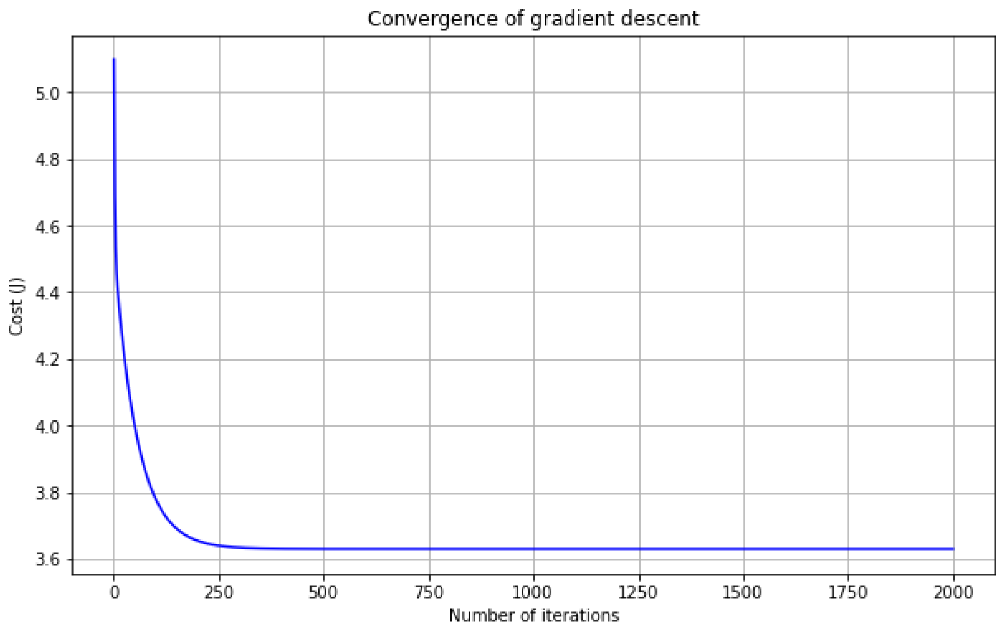

Figure 6: Loss (J) over the number of iterations for X3.

Based on the result obtained from the model, explanatory variables X1 has the lowest loss for explaining the output and the loss value is 0.985.

For larger values of alpha (e.g., 0.1), the gradient descent algorithm convergence in fewer training iterations but the loss curve over number of iterations is not smooth. On the other hand, for smaller values of alpha (e.g., 0.01), the algorithm takes more iterations to converge but the loss curve looks smooth. In addition, with a smaller value of alpha, the gradient descent algorithm approaches the global minimum at a slower rate, which in this case it desirable since it is taking enough iteration to learn all the necessary trends in the training set.

Problem 2.

In this exercise, a linear regression model with gradient descent algorithm with multiple (three) explanatory variables as input was developed and the following results were obtained.

The linear regression model was trained with the following parameters:

- An alpha value of 0.04,
- Number of iterations of 2000,
- theta vector initialized to (0., 0.).

Based on the model finale results, we obtained:

- Final theta vector: **[5.312, -2.003, 0.533, -0.265],** rounded to three decimal places.
- Final loss value of approximately **0.738**, rounded to three decimal places.

Figure 7 shows the loss over the number of iterations for all three explanatory variables.

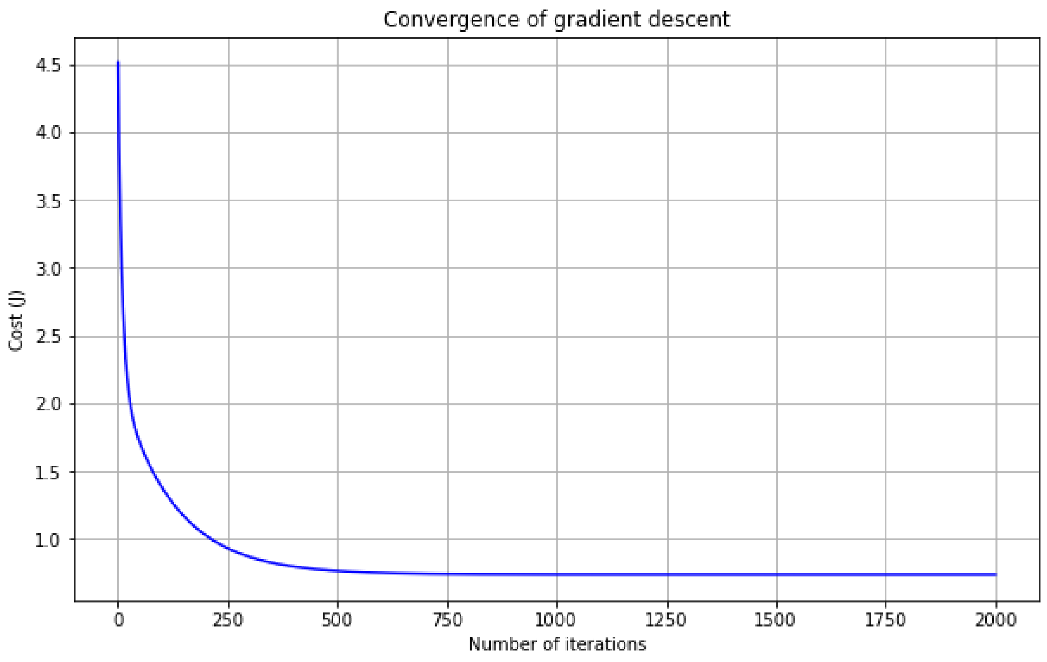

Figure 7: Loss (J) over the number iterations for all three variables.

Based on the training observations, for larger values of the learning rate the gradient descent algorithm converges to the global minimum in fewer iterations, whereas, for smaller values of the learning algorithm takes more iterations to converge to the global minimum.

Learning rate, alpha	**Final loss value**
0.02	0.73895426
0.04	0.73846441
0.06	0.73846424
0.08	0.73846424
0.1	0.73846424

Table 1: Learning rate and loss values for a 2000 number of iterations

As shown in table 1, for larger values of learning rate (e.g., 0.04) the loss converges to its final values well before the 2000 training iterations.

Predicted values of y for the following test inputs,

- Input: (1, 1, 1) -> **Output: 3.547** (rounded to three decimal places).
- Input: (2, 0, 4) -> **Output: 0.244** (rounded to three decimal places).
- Input: (3, 2, 1) -> **Output: 0.102** (rounded to three decimal places).