Claude Shyaka
ID#: 801326243
ECGR-5105

Homework 1: Linear Regression with Gradient Descent Algorithm

GitHub Repo: https://github.com/claudeshyaka/ml

In this exercise, a linear recession model with gradient descent algorithm was developed and used to estimate the price of houses. The US Housing dataset was used to train the model. 80% of the data was used for training and 20% was used for validation.

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished

Table 1: Preview from of the dataset.

1. Without using Normalization/standardization
    a. In this section, housing prices were predicted based on the following input variables: area, bedrooms, bathrooms, stories, parking. After training and evaluation with a **learning rate of 1x10^{-9}** and **1000 iterations**, the following result were obtained:
        i. **Final theta vector: [0.525, 837.930, 1.835, 0.941, 1.438, 0.413]**
        ii. **Final training loss: 1.589x10^{12}**
        iii. **Final validation loss: 2.034x10^{12}**

    Figure 1 shows the training and validation loss curve for 1000 iterations.

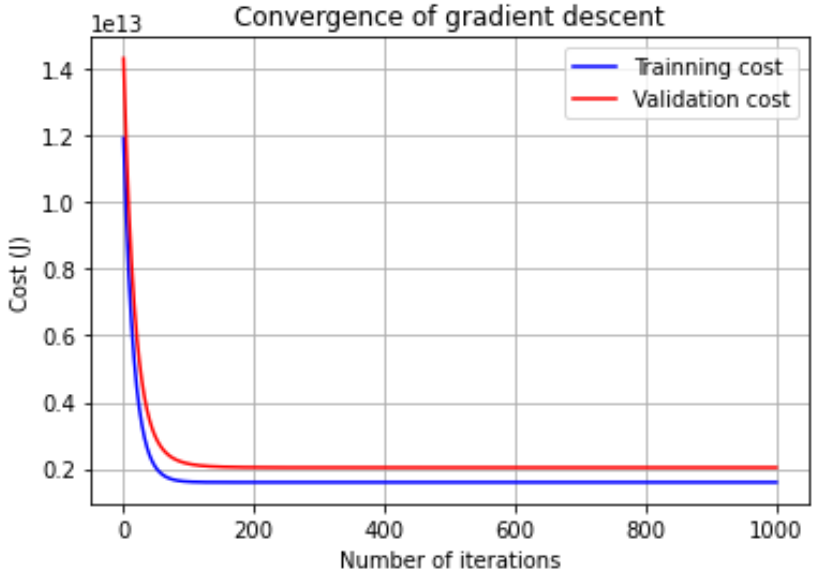

Figure 1

    b. Next, the following input variables were used: area, bedrooms, bathrooms, stories, mainroad, guestroom, basement, hotwaterheating, airconditioning, parking,

prefarea. After training and evaluation with a **learning rate of 1x10^{-9}** and **1000 iterations**, the following result were obtained:

i. **Final theta vector: [0.525, 837.929, 1.835, 0.941, 1.438, 0.455, 0.151, 0.327, 0.057, 0.365, 0.413, 0.189]**
ii. **Final training loss: 1.589x10^{12}**
iii. **Final validation loss: 2.034x10^{12}**

Figure 2 shows the training and validation loss curve for 1000 iterations.

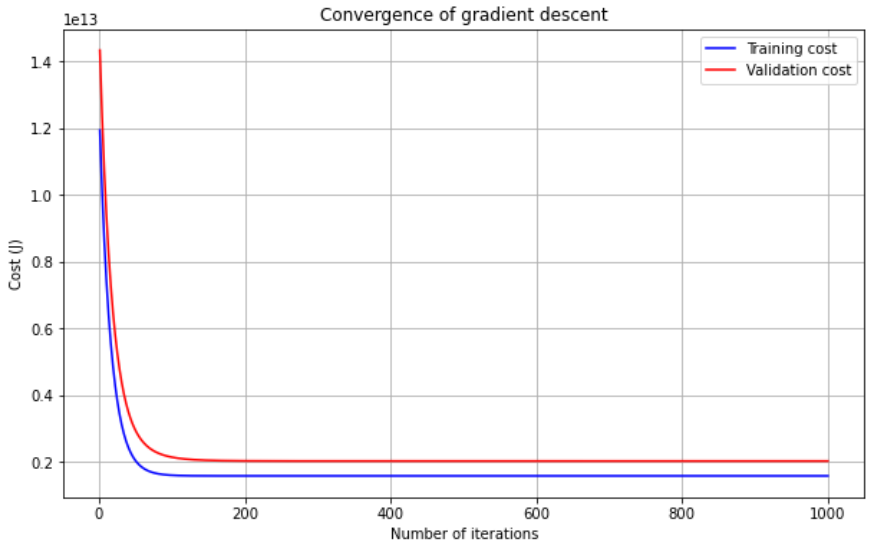

Figure 2

2. Using Normalization/Standardization
    a. For input variables: **area, bedrooms, bathrooms, stories, parking**. As we can see below scaling the input data significantly improves the model performance on both training and validation sets. Also, the MinMaxScaler achieved better training result. Here is a summary of results were obtained.
        i. Using the **MinMaxScaler** to normalize the training and validation data and setting the **learning rate to 0.07** for **5000 iterations**.
            - **Final theta vector: [0.061, 0.428, 0.073, 0.338, 0.141, 0.097]**
            - **Final training loss: 0.00612**
            - **Final validation loss: 0.01076**

        Figure 3 shows the training and validation loss curve for 5000 iterations.

        ii. Using the **StandardScaler** to standardize the training and validation data and setting the **learning rate to 0.07** for **200 iterations**.
            - **Final theta vector: [-2.325e-16, 3.873e-01, 6.438e-02, 3.219e-01, 2.416e-01, 1.642e-01]**
            - **Final training loss: 0.2189**
            - **Final validation loss: 0.2117**

        Figure 3 shows the training and validation loss curve for 200 iterations.

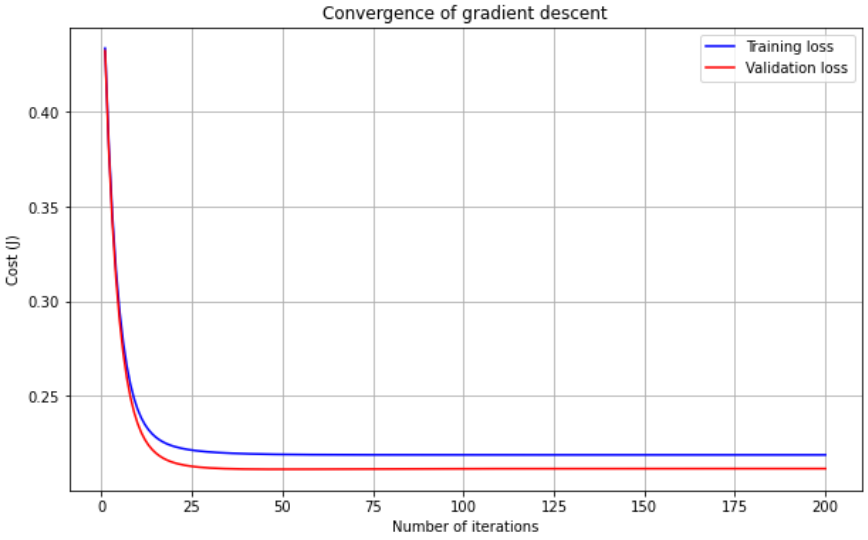

Figure 3

b. For input variables: **area, bedrooms, bathrooms, stories, mainroad, guestroom, basement, hotwaterheating, airconditioning, parking, prefarea**. As we can see below scaling the input data significantly improves the model performance on both training and validation sets. Also, the MinMaxScaler achieved better training result. Here is a summary of results were obtained.

   i. Using the **MinMaxScaler** to normalize the training and validation data and setting the **learning rate to 0.05** for **10000 iterations**.

      - **Final theta vector: [0.0175, 0.3293, 0.0376, 0.3169, 0.1219, 0.0393, 0.0232, 0.0414, 0.0678, 0.0769, 0.0707, 0.0606]**
      - **Final training loss: 0.00450**
      - **Final validation loss: 0.00865**

   Figure 4 shows the training and validation loss curve for 10000 iterations.

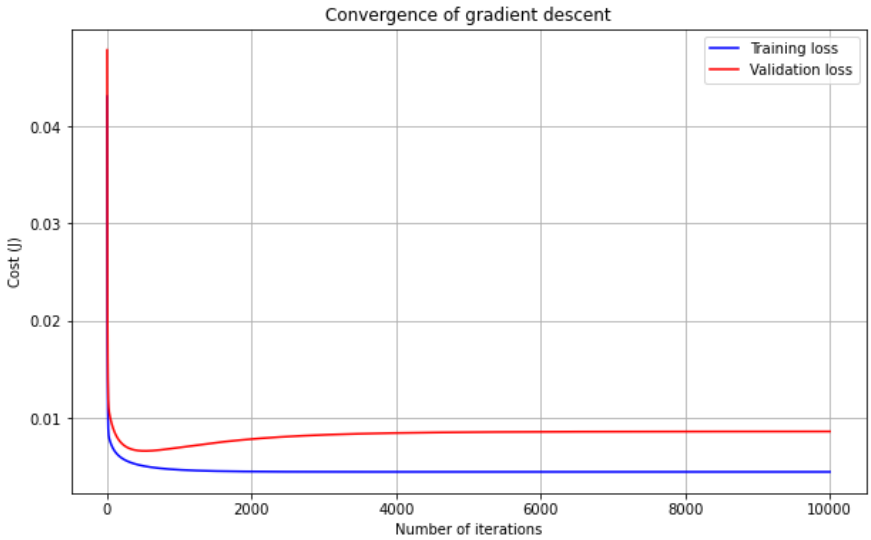

Figure 4

   ii. Using the **StandardScaler** to standardize the training and validation data and setting the **learning rate to 0.01** for **3000 iterations**.

- **Final theta vector: [-2.118e-16, 2.981e-0.1, 3.338e-02, 3.012e-01, 2.084e-01, 8.208e-02, 5.317e-02, 1.187e-01, 8.879e-02, 2.124e-01, 1.203e-01, 1.535e-01]**
- **Final training loss: 0.1609**
- **Final validation loss: 0.1604**

Figure 5 shows the training and validation loss curve for 3000 iterations.

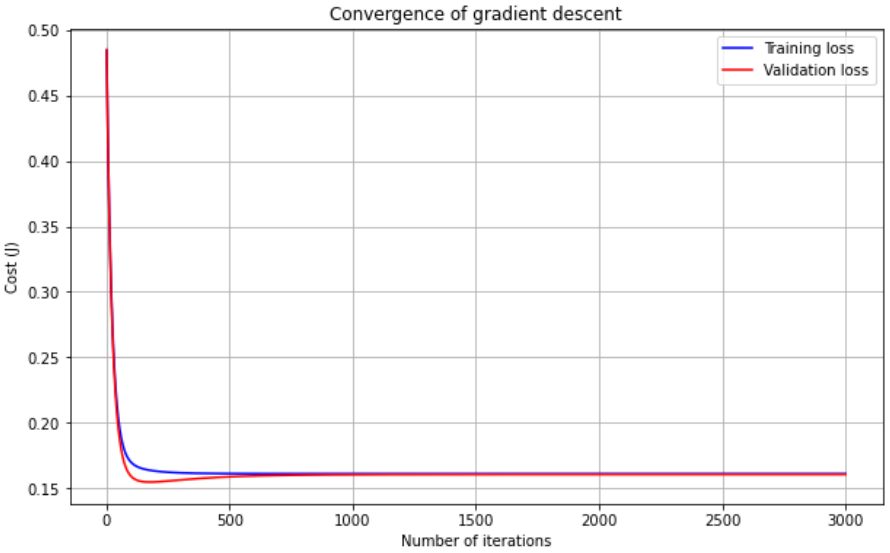

Figure 5

3. Using Normalization/Standardizations and adding Parameter Penalization to improve model performance on validation set.
    a. For input variables: **area, bedrooms, bathrooms, stories, parking**. After training and validating the model with both MinMaxScaler and StandardScaler applied to the input data, results showed that the model performed better when the MinMaxScaler was used. Using the MinMaxScaler to normalize the training and validation data and setting the **learning rate to 0.01** for **5000 iterations** with a **lambda value of 4**, the following results were obtained.
        - **Final theta vector: [0.0962, 0.2522, 0.0975, 0.2186, 0.1463, 0.1278]**
        - **Final training loss: 0.00737**
        - **Final validation loss: 0.00917**

    Notice that the validation loss in lower and training loss is a bit higher compared to result from problem 2.a.i. This could suggest that the model was overfitted to the training data in problem 2.a.i. Overall, results obtained here (problem 3.a) seem more promising to perform better on unseen data compared to results in problem 2.a.i.

    Figure 6 shows the training and validation loss curve for 5000 iterations.

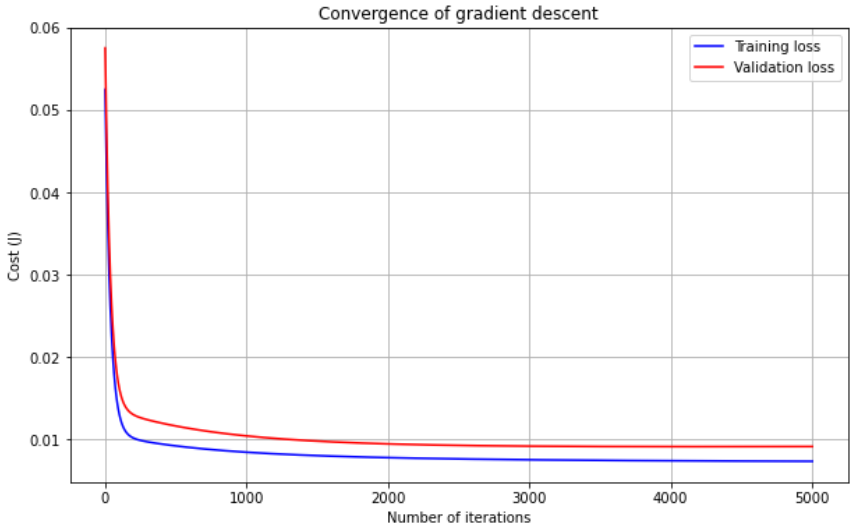

Figure 6

b. For input variables: **area, bedrooms, bathrooms, stories, mainroad, guestroom, basement, hotwaterheating, airconditioning, parking, prefarea**. After training and validating the model with both MinMaxScaler and StandardScaler applied to the input data, results showed that the model performed better when the MinMaxScaler was used. Using the MinMaxScaler to normalize the training and validation data and setting the **learning rate to 0.02** for **2000 iterations** with a **lambda value of 4**, the following results were obtained.

- **Final theta vector: [0.0377, 0.1686, 0.0717, 0.1835, 0.1190, 0.0492, 0.0345, 0.0374, 0.0570, 0.0871, 0.0966, 0.0629]**
- **Final training loss: 0.00548**
- **Final validation loss: 0.00674**

Notice that the validation loss in lower and training loss is a bit higher compared to result from problem 2.b.i. This could suggest that the model was overfitted to the training data in problem 2.b.i. Overall, results obtained here (problem 3.b) seem more promising to perform better on unseen data compared to results in problem 2.b.i.