**COMP-551 : Applied Machine Learning Given on: Jan 16, 11pm Programming Assignment #1 Due on : Jan 29, 12pm**

Student’s name: Frederic Ladouceur Student’s ID: 260690491

**1.**

**1.1** The training MSE is 6.47679853116 and the validation MSE is 1419.91389547. The following plot shows how the linear polynomial regression fits the training data points. Some noise is present in our data. The fitting curves tries to accommodate all the training data points and minimizes the MSE for this dataset. However, the curve is unstable: it wiggles a lot and drops drastically as it approaches 1 and -1(results of overfitting). This indicates us that this curve, while having a low training error, might not be the best regression curve for predicting this kind of data (as we can see by looking at the very large testing MSE).

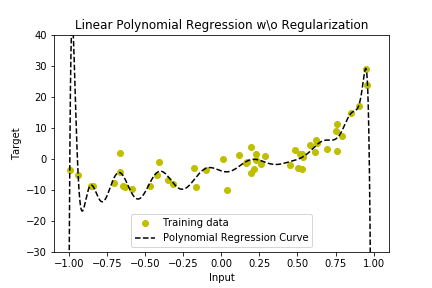


Figure 1: Linear Polynomial Regression without Regularization

**1.2** The first 2 following plots respectively show how the training and the validation MSE vary with λ going from 0 to 1. The validation data tells us that the λ that gives us the smallest MSE is 0.0197 (which corresponds to a MSE of 9.13508347301 on the validation set). The third plot shows the fit of the polynomial regression curve on the testing data. Its MSE is 10.7323010053. Since it is very close to the minimal validation MSE, we deduct that this curve predicts the data very well. The fourth graph shows us the fit of the regression curve on the training dataset. The MSE is 8.85645680048 which is close to the training MSE found in 1.1, but now the curve seems more stable. It is smooth and does not drop at -1 and 1. It now shows better performance at predicting unseen data points (when testing).

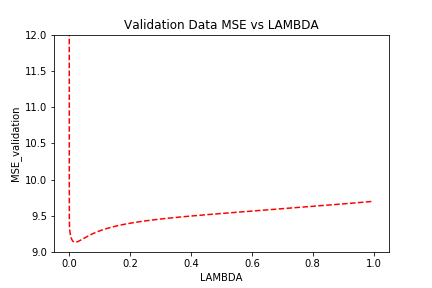
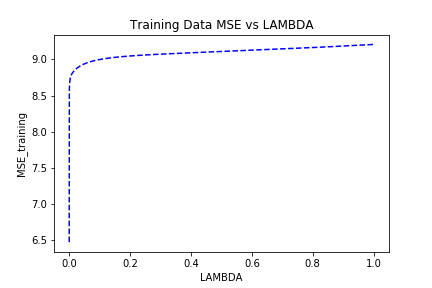


Figure 2: Training Data MSE vs LAMBDA

Figure 3: Testing Data MSE vs LAMBDA

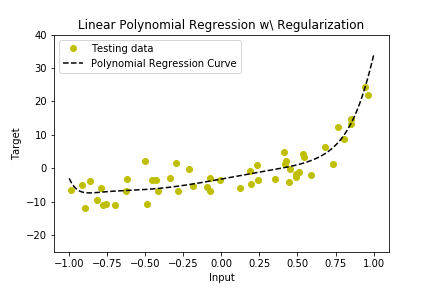


Figure 4: Linear Polynomial Regression with Regularization on Testing Data

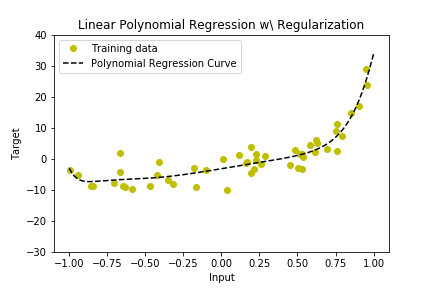


Figure 5: Linear Polynomial Regression with Regularization on Training Data

**1.3** By looking at the above graph, I would say that the data describes a polynomial of second degree since the shape of its regularized regression curve is similar to a parabola.

**2.**

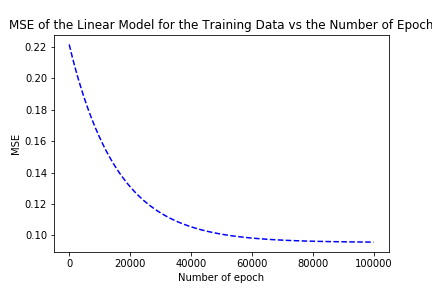
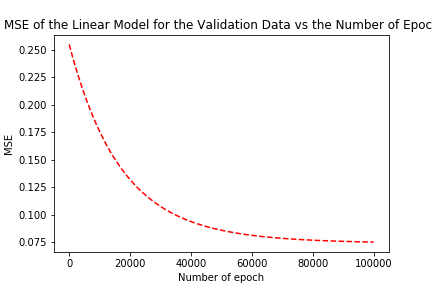
**2.1** I fitted the linear regression model on the training data points using online-SDG with parameters having starting values: The w0=4.3, w1=3.5. The first 2 following plots respectively show how the training and the validation MSE vary with the number of epochs varying from 0 to 100,000. We see that the MSE only decreases as the number of epochs grows. The third plot show the fit of a linear regression model on the training data.

Figure 6: MSE for Validation Data vs Number of Epochs

Figure 7: MSE for Training Data vs Number of Epochs

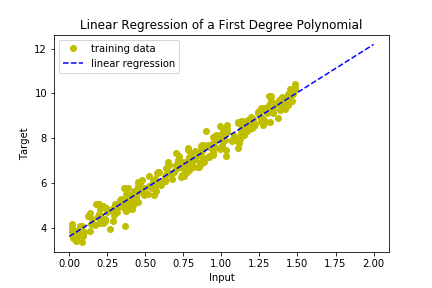


Figure 8: Online-SDG with 100k epochs and 1.0e-06 step size

**2.2** Since the code is heavy and takes a long time to process I manually picked different step size values instead of making it vary “continuously” as I did previously. The following table shows how the validation MSE varies with the step size growing logarithmically.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **The Validation MSE for Different Step Sizes** | | | | | | | |
| Step size | 1.00E-07 | 1.00E-06 | 1.00E-05 | 1.00E-04 | 1.00E-03 | 1.00E-02 | 1.00E-01 |
| Lowest validation MSE | 0.175881 | 0.075335 | 0.0738879 | 0.073879 | 0.073798 | 0.073824 | 0.077094 |

The step size that gives the lowest MSE is 1.00E-04. The test MSE for this step size at 100k epochs is 0.0687187132877.

**2.3** I applied my online-SDG method on the training dataset and computed the lowest validation MSE with 5 different number of epochs. Below are the table of results and the plots showing the line fitting for the five different experiments (from 10 epochs, being the first graph, to 100000 epochs, being the last graph). We see that the fit gets better and better as the number of epochs grows (the MSE decreases).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **The Validation MSE for Different Numbers of Epochs** | | | | | |
| Number of epochs | 10 | 100 | 1000 | 10000 | 100000 |
| Lowest Validation MSE | 0.254928 | 0.253975 | 0.2447822 | 0.175881 | 0.075335 |

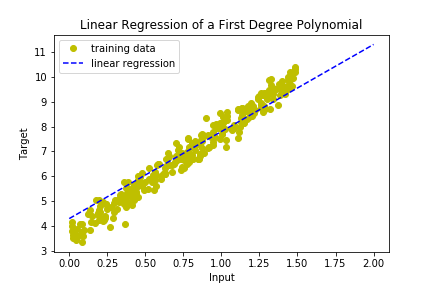


Figure 9: Online-SDG going through 10 epochs

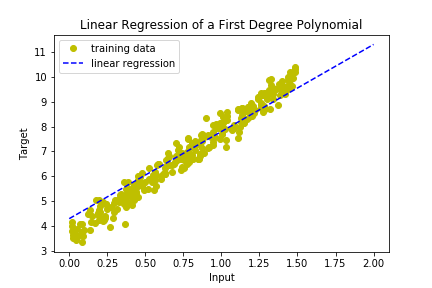


Figure 10: Online-SDG going through 100 epochs

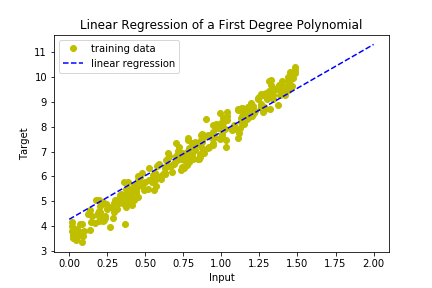


Figure 11: Online-SDG going through 1000 epochs

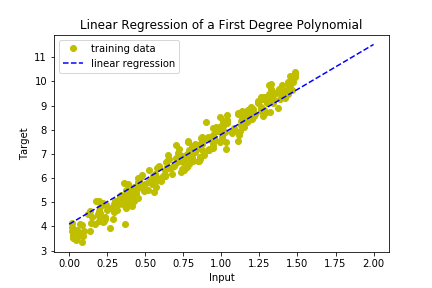


Figure 12: Online-SDG going through 10000 epochs

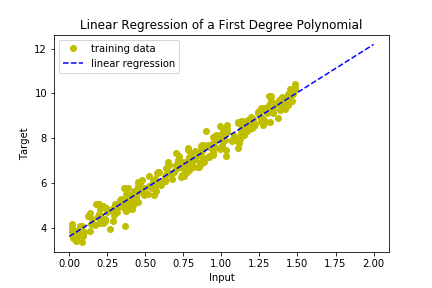


Figure 13: Online-SDG going through 100000 epochs

**3.**

**3.1** Filling the missing pieces of data with the feature mean is a good choice since there are many features (high-dimensionality) and it would be tedious to find a better way to fill each column. Additionally, this technique is widely used in data mining. I could have used the median instead or I could have fill the missing elements using a random function having the same mean and STD as the column. I could also delete all the rows that have missing elements. The completed dataset is stored under the following path: “out/3.1/Dataset\_3\_meanfill.csv”.

**3.2** The 5-fold cross-validation error is 0.01580035 and the parameters learnt are stored under the path: “out/3.2/3.2\_w\_star.csv”

**3.3** I tested my algorithm with values of λ ranging from 0.0 to 5.0 with increments of 0.05 and I produced the following graph with my values. The first plot shows how the MSE varies with the value of λ for the training data and the second graph shows how the MSE varies with the value of λ for the testing data. The λ that gives me the smallest testing MSE is 2.105 and the minimal MSE is 0.0172061085405. This LAMBDA allows me to compute the best-solution parameters which are stored under the path: “out/3.3/3.3\_w\_star.csv”. Feature selection could be achieved by looking at the weights of the parameters in the file 3.3\_w\_star.csv. The weights having the largest absolute values correspond to the features with the most importance. I reduced my set of features by half by only selecting the features that had an absolute weight above the median of the absolute weights. In that way, only the most prominent features are kept. I trained the algorithm using only one reduced data set and no regularization term. The MSE on the testing data was 0.01935828 which is very close to what I had before. This indicates us that the reduced feature set might be a good selection considering the fact that no regularization was applied.

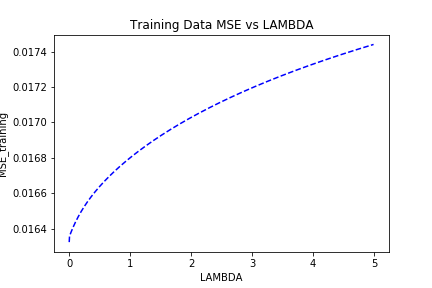
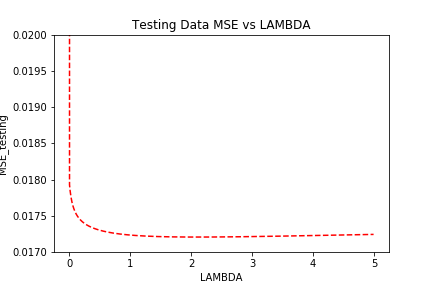


Figure 14: Testing MSE vs LAMBDA

Figure 15: Training MSE vs LAMBDA

**References**

-I discussed with Frank Ye (section 2, ID: 260689448) about this assignment. We compared our

evaluation techniques.