**ICS 485**



**ASSIGNMENT #2**

**Term 191**

Department of Computer Science and Engineering

King Fahd University of Petroleum and Minerals

Dhahran, Eastern Province, Saudi Arabia - 31261

19/10/2019

**Machine Learning**

Submitted by

**Team 4**

Under the supervision of

**Dr. Irfan Ahmed**

# INTRODUCTION

In this assignment we were introduced with two sets of data to work on using least squares regression. The Least Squares Regression Line is the line that makes the vertical distance from the data points to the regression line as small as possible. To find the best fit line we will use different regression models and analyze the results.

Starting with the first data set that represent **diabetes** data collected by Stanford, we implemented the closed form solution using W = (

And the iterative solution using gradient descent.

For the second part we worked with the second data set that represent both X and Y in this function y = wx + e. First, we implemented the gradient descent for **Ridge** regression which takes L2 as its regularization method, then the same way we implemented gradient descent for **Lasso** regression using L1 as its regularization method.

L1=

L2=

Finally we compared between our implementation and the SkLearn built in implementation of both Ridge and Lasso regression models.

# PARTICIPATION

|  |  |
| --- | --- |
| Name | Percentage |
| Baraa Fael | 40% part A |
| Faris Hijazi | 35% part B implementation |
| Mohammed Al Sayed | 25% report + part B testing + answering |

# QUESTIONS:

## **Part A:**

1. Compare your results for the three approaches, i.e., using library, using closed-form solution, and using the iterative solution. Provide your comments on the results.

computing the training MSE and test MSE when fitting a regressor to all features with a data set of **20**

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Gradient descent (iterative) | Closed form solution | Built-in linear regression |
| Training | 4822 | 1487 | 1636 |
| Testing | 126975 | 129749 | 6764 |

computing the training MSE and test MSE when fitting a regressor to all features with a data set of **50**

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Gradient descent (iterative) | Closed form solution | Built-in linear regression |
| Training | 11272 | 5487 | 2414 |
| Testing | 117077 | 142389 | 7991 |

computing the training MSE and test MSE when fitting a regressor to all features with a data set of **100**

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Gradient descent (iterative) | Closed form solution | Built-in linear regression |
| Training | 24318 | 13108 | 2883 |
| Testing | 104172 | 55699 | 3583 |

computing the training MSE and test MSE when fitting a regressor to all features with a data set of **200**

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Gradient descent (iterative) | Closed form solution | Built-in linear regression |
| Training | 54403 | 25989 | 2858 |
| Testing | 73918 | 33313 | 3028 |

computing the training MSE and test MSE when fitting a regressor to all features with a data set of **300**

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Gradient descent (iterative) | Closed form solution | Built-in linear regression |
| Training | 81072 | 39616 | 2905 |
| Testing | 47240 | 18575 | 2877 |

Analysis:

Starting with the 20 data set we can see that the built-in approach is much better than both the iterative and the closed form solution (Training and Testing).The iterative and the closed form has close results in testing but they differ in training where the closed form get better results.

Using 50 data set we see that the built-in approach is still giving better results that both the iterative and the closed form but the gap between them is larger. The iterative get’s higher results than the closed in training but not in training where it shows a better fit than the closed.

1. Compare the parameter values for the three solutions when using n\_train = 300 training samples.

Checking all the approaches we can see that the best result for testing is using the **300** data set and the MSE for each approach is as follows:

Built-in approach MSE = 2877

Closed form solution MSE = 18575

Iterative approach MSE = 47240

*Using 100 as the iteration number and 0.0000001 as the learning rate*

1. What changes you need to do if the unit of y is different?

Assuming the change is only in the scale, there are two options:

* 1) Don’t change anything. In this case, the **values** of all associated weights are going to be scaled up or down based on the scale of Y, and only the interpretation of the Y unit is going to be different
* 2) Before you insert the data in the model, rescale all features’ units in accordance with the scale of Y. In this case, the **values** of the weights are going to stay as is but with different interpretation (units)

1. What changes you need to do if the unit of one of the features was different? For example, if age was in months and not in years.

No changes needed. However, only the weight corresponding to that feature is going to have a different unit (interpretation). The slope is always the change in the response variable (in whatever unit that is measured in) for a unit change in each predictor variable - for whatever unit that is measured in.

1. What if both 3 and 4 apply?

1) Scale all features’ units correspondingly. In this case, weights are going to have the same values with different units except for the feature that we changed its scale, too.

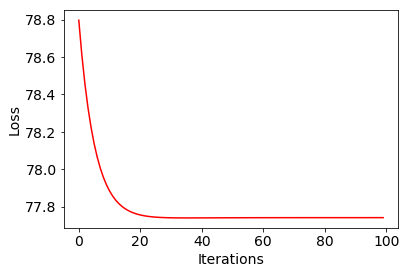
2) Don’t change anything. In this case, all weights are going to have different values and different y unit interpretation.

Note: The X that we have changed its unit is excluded from both cases since it is going to suffer a change in its both X and Y components, hence, a completely different interpretation and unit. However, in all cases the change in response per change in unit feature will **relatively** stay the same

## **Part B:**

1. Documents all the results in the report

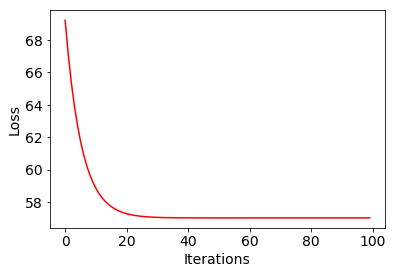
1. Gradient descent solver for ridge regression with c=1

final loss= 77.74109430643726

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Ridge gradient descent solver | Built-in gradient descent solver | Built-in linear regression |
| Training | 51.50863 | 0.034804 | 1.14229e-28 |
| Testing | 52.62271 | 4.59016 | 50.83412 |

2. Lasso solver for ridge regression with c=1

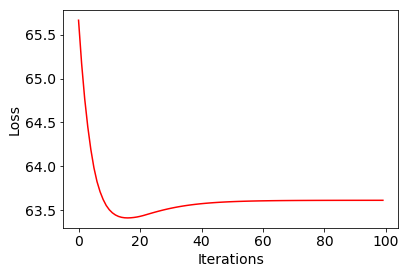
final loss= 57.02033563734428



|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Lasso solver | Built-in gradient descent solver | Built-in linear regression |
| Training | 59.65430 | 8.78299 | 1.14229e-28 |
| Testing | 51.65272 | 9.56604 | 50.83412 |

1. Try with a large value of C (e.g., 20) for lasso and check the weights and MSE. What do you observe?

final loss= 63.61399793558083



training MSE: 47.184663045189076

training MSE: 50.215260389855786

we notice that it gives better results for both the training and testing, and that it takes more iterations to converge.

1. Compare the coefficients (parameter values) for ridge and lasso for the best setup. What do you observe? Can you explain?

Comparing values magnitude of the weights (||W||), we can see the following:

|  |  |  |
| --- | --- | --- |
| Ridge solver | Lasso solver | Non-regularized solver |
| 83.21 | 89.22 | 83.20 |

We are noticing that the lasso solver has a larger magnitude, and that’s because the lasso regularization term is not squared like the ridge regularization, meaning it would punish less for high weights (thus allowing for higher weights).

In terms of why the weights themselves are so close, I’m not sure, maybe the random seed for the weight initialization is not indicative, but the weights should be much higher without regularization (in theory). Random seed *random\_state=9* was used for these tests.

1. Compare MSE of linear, ridge, and lasso. What do you observe?

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Model | | |
| Lasso solver | Ridge solver | Built-in linear regression |
| Training | 59.65430 | 51.50863 | 1.14229e-28 |
| Testing | 51.65272 | 52.62271 | 50.83412 |

we notice that the built in has the better results in training by a large difference, but in testing they all have close values of MSE.

1. Which among the ridge and lasso gives the best results on the test? Can you explain why?

Ridge regression gave better results (lower error) when testing.

In theory, I expected the lasso to do better because it punishes less, thus lasso should be more encouraging for the model to learn, however this may be another issue with a bad random seed.

1. Can the lasso regression retrieve the 10 features which were used in the equation for y? List them.