CSE574 Spring 2019 Introduction to Machine Learning
Programming Assignment 1
Classification and Regression

Group Members:

Naga Vaishnavi Pakyala
Bhanu Prasanth Yeeli
Siddiq Syed

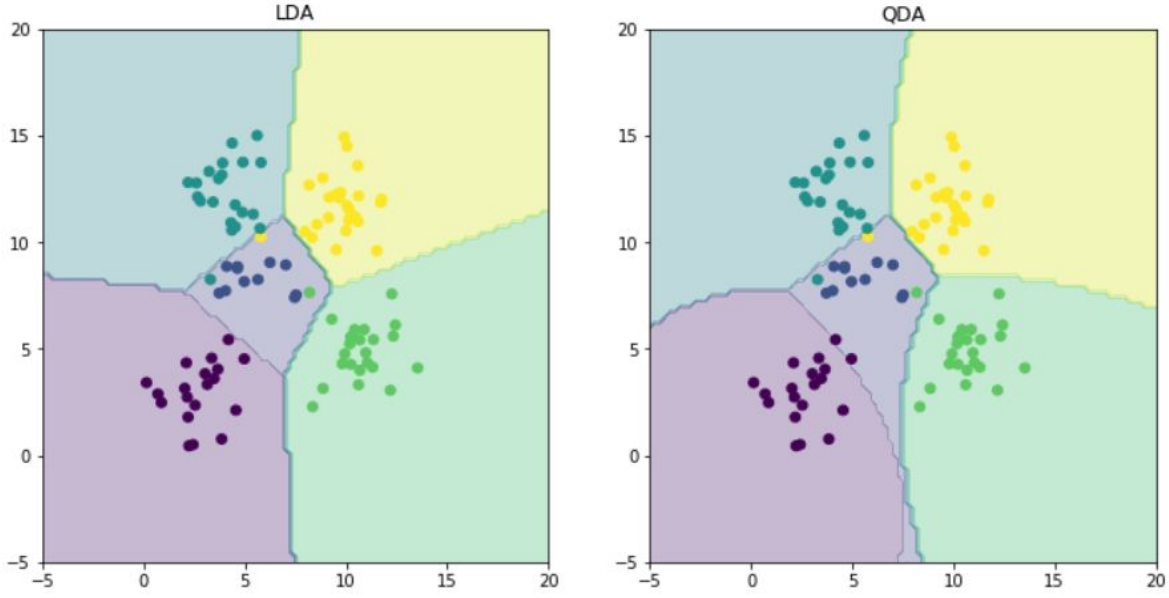
Problem 1:

The aim is to experiment with Gaussian discriminants by implementing Linear discriminant analysis and Quadratic discriminant analysis. The ldalearn and qdalearn considers the training data set which a 2D sample data set resulting in computing the mean and covariance matrix.

LDA and QDA Accuracies:

```
LDA Accuracy = 0.97
QDA Accuracy = 0.94
```

Plots for Linear and Quadratic Discriminators:



Inference:

- From the plot we can visualize that both the plots **does not have similar boundaries** and also from the accuracies above it is clear that LDA and QDA has **different accuracies**.

- It is apparent that **LDA**(Linear discriminant analysis) has linear boundaries and **QDA**(Quadratic discriminant analysis) has curved boundaries.
- The accuracies of LDA and QDA are quite close, QDA has little **less accuracy** than LDA because of its **curved boundaries**.
- The class in the center of the figure of QDA has more number of **miss classification** when compared to LDA.

Problem 2:

The aim is to implement ordinary least squares method to estimate regression parameters by minimizing the squared loss and implement the function testOLERegression to apply the learnt weights for prediction on both training and testing data and to calculate the mean squared error (MSE).

Calculated Mean Squared error:

```
MSE without intercept for train data[19099.44684457]
MSE with intercept for train data[2187.16029493]
MSE without intercept for test data [106775.36155223]
MSE with intercept for test data [3707.84018148]
```

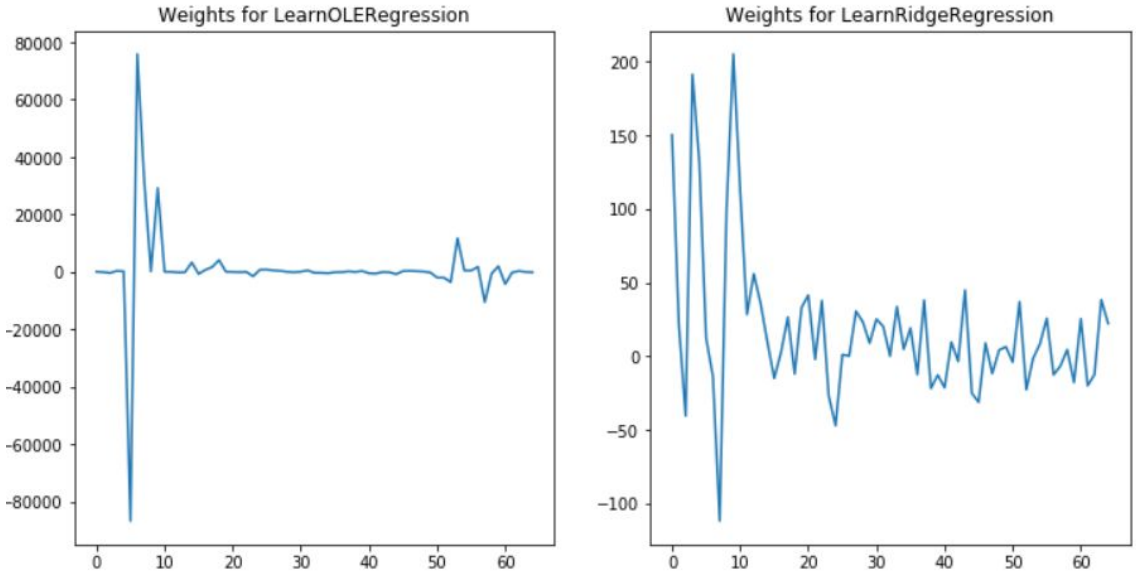
Inference:

- As this is a comparison for error, lower the error better will be the accuracy of **Linear Regression**.
- Here, while comparing the performances of all the cases, the MSE with the **intercept** performs better than the MSE **without the intercept**.
- Also, MSE with and without intercept performs better on the Training data than on the Test data, the reason being less amount of **testing data**.

Problem 3:

The aim is to implement parameter estimation for ridge regression by minimizing the regularized squared loss.

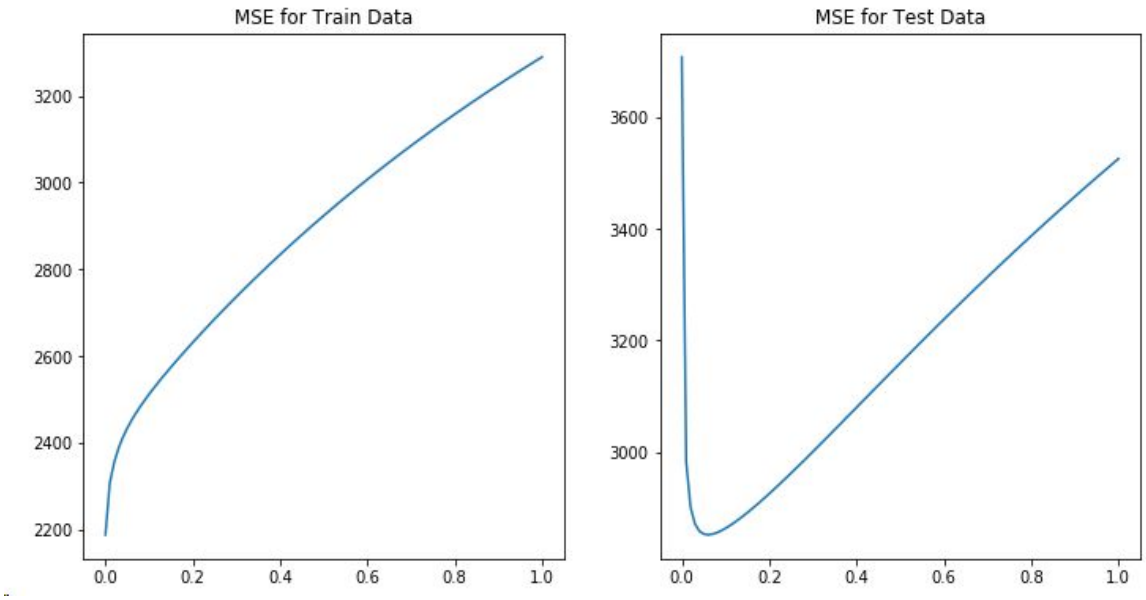
Comparing the relative magnitudes of weights learnt using OLE and weights learnt using ridge regression:



Inference:

- The initial values for **OLE** are too high and slowly they were ossicalting around 0. The **Ridge** weights as pretty much close to zero compared to OLE.

Plot for the errors on train and test data obtained without using the gradient descent:



Calculated Mean Squared error with learnOLERegression:

```
MSE without intercept for train data[19099.44684457]
MSE with intercept for train data[2187.16029493]
MSE without intercept for test data [106775.36155223]
MSE with intercept for test data [3707.84018148]
```

```
Minimum Error of Ridge Regression at lambda :0.06
```

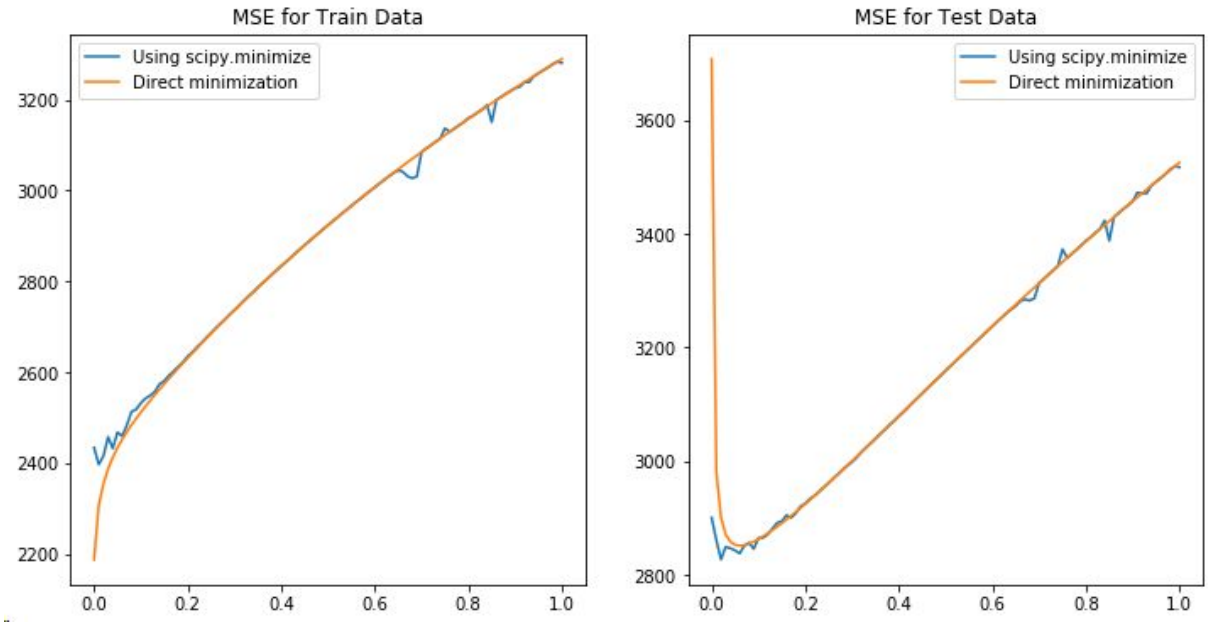
Inference:

- The both train and test MSE is increasing as the lambda goes from no regularization to regularization. Also, the test MSE is always greater than the train MSE which is apparent.
- With intercept MSE of OLE is almost close to the no regularization error of the Ridge. But then the test MSE of Ridge is less when compared to test error of OLE.
- Hence the optimal lambda value can be chosen when test error of the Ridge is lowest i.e. at lambda = 0.06 from code screenshot above.

Problem 4:

The aim is to implement the gradient descent procedure for estimating the weights. Also, using the minimize function and regressionObjVal, compute the regularized squared error and its gradient with respect to w.

Plot for the errors on train and test data obtained by using the gradient descent:



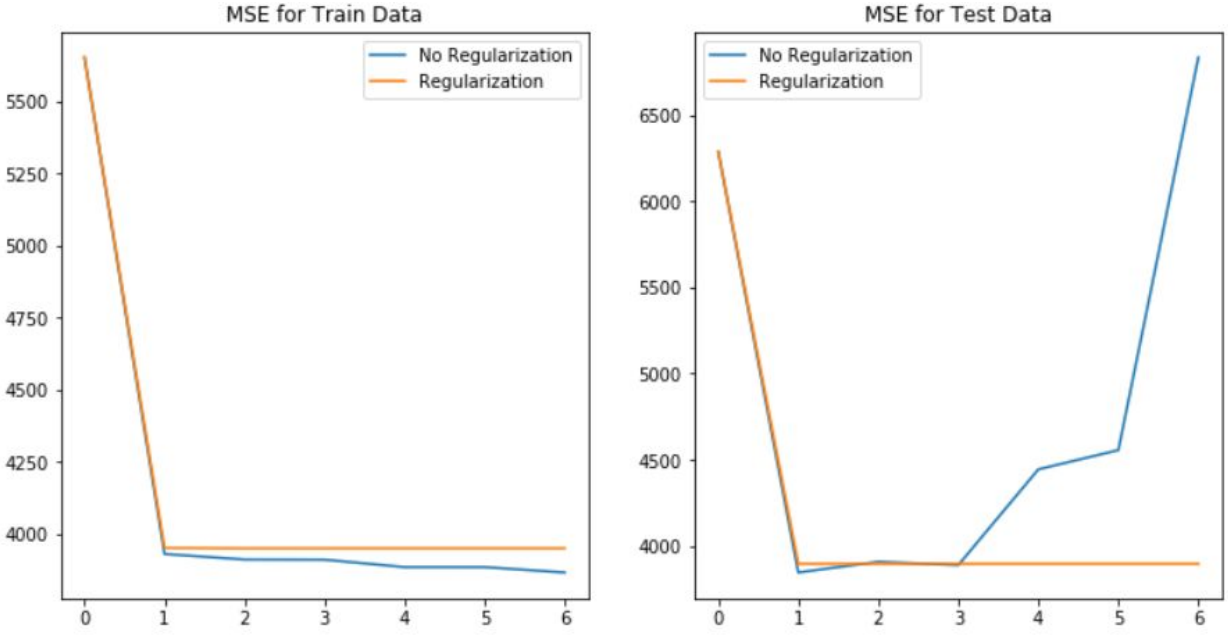
Inference:

- It can be inferred from the plots that **blue lines** are retrieved by using **scipy.minimize** and **orange line** is retrieved by using **Direct minimization**.
- It can be observed that for the **training data**, the error is directly proportional to the value of lambda and particularly in this plot, error increases as lambda increases. Even upon usage of scipy.minimize, the error increases with the value of lambda.
- It can be observed that for the **test data**, the error is inconsistent. It is noted to decrease until a certain value of lambda and then increase with increase in lambda. Even upon usage of scipy.minimize, the error increases after a certain value of lambda
- In comparison with **Problem 3**, it can be analysed (with and without gradient descents) that the **Minimum MSE is minimized** on the train and test data obtained by **using the gradient descents**. But on the whole, based on the **overall performance**, the curves obtained on the plots are more or less similar.

Problem 5:

The aim is to investigate the impact of using higher order polynomials for the input features. For this problem, we use the third variable as the only input variable to analyse the results. Additionally, implement the function mapNonLinear which converts a single attribute x into a vector of p attributes.

Plots of MSE with and without regularization for test and train data:



Inference:

- It can be inferred from the plots that **blue line** is MSE that is retrieved by using **No Regularization** and **orange line** is MSE that is retrieved by using **Regularization** for p through 0 to 6.
- It can be observed that for the **training data** when no regularization is performed**,** for the values of p through 0 to 6, there is a consistent decrease of MSE.
- It can be observed that for the **test data** when no regularization is performed**,** for the values of p through 0 to 6, there is an increase of MSE.
- It can be observed that for the **test and training data**, upon regularization, the MSE remains constant.
- While comparing the optimal value of p in terms of test error for both settings, it is observed that with regularization the optimal value of p any value greater than 1 as MSE remains constant and without regularization the optimal value of p would be 1.

Problem 6:

Final Observations

Training data:

MSE with intercept of Linear Regression: **2187.16029493**

MSE without intercept of Linear Regression: **19099.44684457**

MSE using Ridge Regression: **2187.16029493 at lambda=0**

MSE using Gradient descent: **2396.44210894 at lambda = 0.01**

MSE with no regularization of Non-Linear Regression: **3866.88344945 at p=5**

MSE with regularization of Non-Linear Regression: **3950.68233514 at p=5**

Test data:

MSE with intercept of Linear Regression: **3707.84018148**

MSE without intercept of Linear Regression: **106775.36155223**

MSE using Ridge Regression: **2851.33021344 at lambda=0.06**

MSE using Gradient descent: **2826.9525654 at lambda=0.02**

MSE with no regularization of Non-Linear Regression: **3845.03473017 at p=1**

MSE with regularization of Non-Linear Regression: **3895.85646447 at p=1**

From above values it is evident that **Gradient descent** has the least MSE value and can be considered the best model to have good fit for test data.