CSE574 Spring 2019 Introduction to Machine Learning
Programming Assignment 1

Group 3

Abhishekh
Nupur
Syed Aqhib Ahmed

March 9, 2019

1. Problem 1: Experiment with Gaussian Discriminators

1.1 Problem Requirements

Im- plement two functions in Python: ldaLearn and qdaLearn which take a training data set (a feature matrix and labels) and return the means and covariance matrix (or matrices). Implement two functions ldaTest and qdaTest which return the true labels for a given test data set and the accuracy using the true labels for the test data.

Train both methods using the sample training data (sample train). Report the accuracy of LDA and QDA on the provided test data set (sample test). Also, plot the discriminating boundary for linear and quadratic discriminators. The code to plot the boundaries is already provided in the base code. Explain why there is a difference in the two boundaries.

1.2 Accuracy of LDA and QDA on the provided test data set

Both the LDA and QDA methods were trained using the same sample data and by testing the models on the test data, we find that the accuracies for both LDA and QDA are the same and equal to 97% on this particular data set.

LDA	0.97
QDA	0.97

1.3 Plotting the discriminating boundaries for LDA and QDA

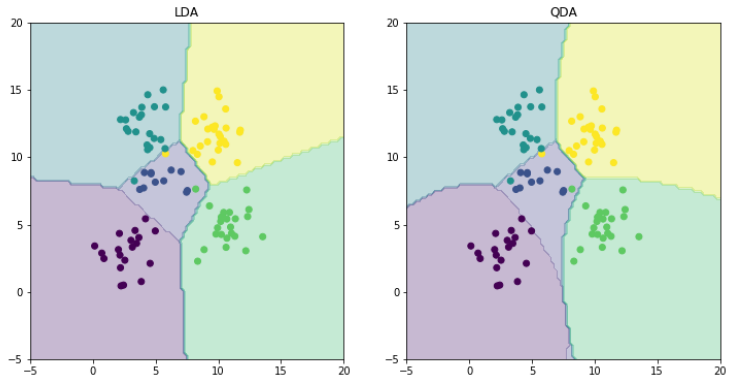

Figure 1: Boundary Plotting

The only significant difference that we notice from the boundary plots of LDA and QDA is that as the name suggests, the LDA model seems to have assigned fairly linear boundaries to the classes whereas the the QDA model has non-linear boundaries for it's classes, which is as expected.

1.4 Reason for Boundary Shapes

The reason for the LDA boundaries to be straight lines is the use of the same covariance matrices for all the classes. But in the case of QDA, the covariance matrices for each class are different, and we notice the presence of quadratic expressions in the likelihood probability calculations, which leads to the class boundaries being non-linear in nature.

It is worth noting that QDA is more susceptible to overfitting if the data can be reasonably bounded by a linear boundary, and hence does not generalize well in those situations.

2. Problem 2: Experiment with Linear Regression

2.1 Problem Requirements

Calculate and report the MSE for training and test data for two cases: rst, without using an intercept (or bias) term, and second with using an intercept. Report on which one is better

2.2 Error Observation

Data	MSE without Intercept	MSE with intercept	Accuracy gain/loss(Ratio)
Train	72219.7833	8142.9660	8.86
Test	333673.0048	11408.7390	29.24

2.3 Comparison of the two methods

From the table of errors above, we notice that introducing a bias (or intercept) has an objectively positive effect on the predictive accuracy of the model. We notice a significant drop in error rates when we introduce bias into the model.

We also notice that the effect is more pronounced on the test set as compared to the training set. Geometrically, this can be explained by the fact that not all real data passes through the origin (assuming we plot the data in n-dimensional space). The bias term is therfore nevessary in order to fit the model to pass through the data more accurately. If we donot include a bias term in our model, we are effectively crippling our model such that it cannot move to fit the data that it is supposed to.

3. Problem 3: Experiment with Ridge Regression

3.1 Problem Requirement

Calculate and report the MSE for training and test data using ridge regression parameters using the the testOLERegression function that you implemented in Problem 2. Use data with intercept. Plot the errors on train and test data for different values of λ.

Vary λ from 0 (no regularization) to 1 in steps of 0.01. Compare the relative magnitudes of weights learnt using OLE (Problem 2) and weights learnt using ridge regression. Compare the two approaches in terms of errors on train and test data. Find the optimal value for λ and explain?

3.2 MSE for train and test dataset using different Lambda values

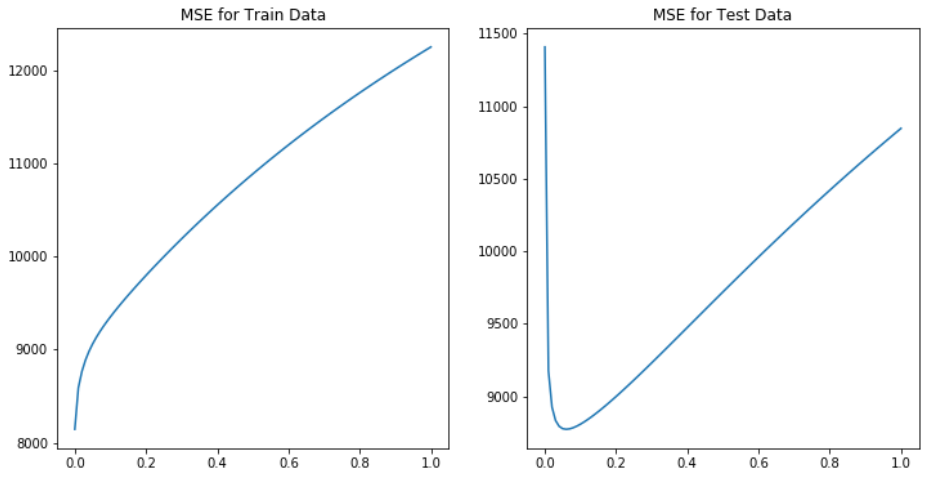

Figure 2: Lambda vs MSE

We notice that the mean squared error generally increases as we increase the value of lambda for both the training data, and for the test data. The MSE's for both train and test have been samples for a few values of lambda and tabulated for reference.

3.3 Relative weights learned by OLE and Ridge

We have plotted the relative weights learned by all of the specified models in Figure: 3.

3.4 Observations

Both ordinary least squares and Ridge regression give the same weight values when

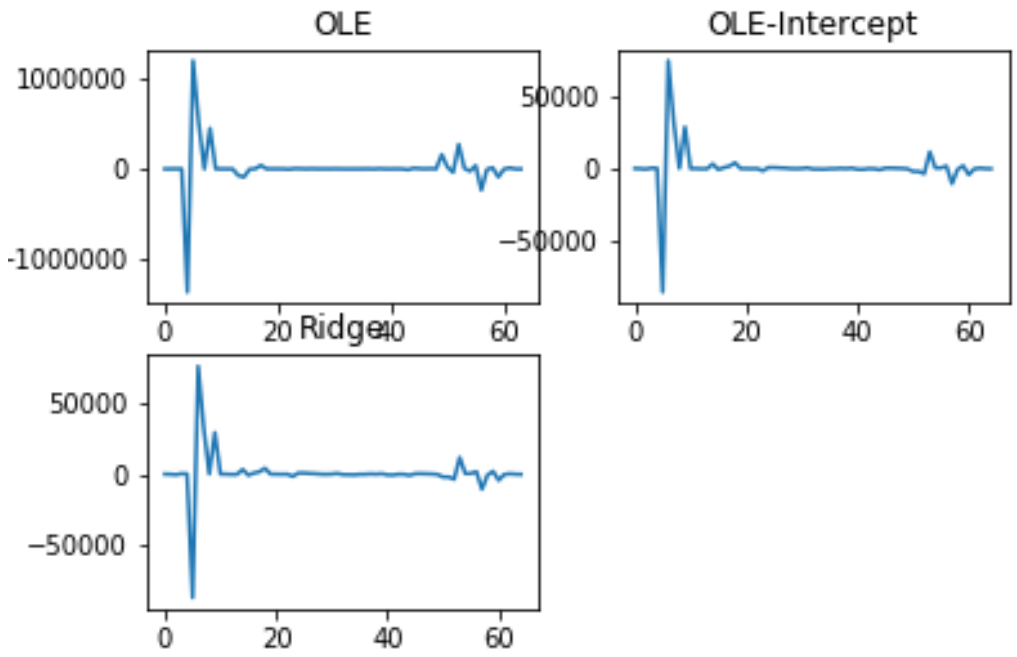

Figure 3: Regression weights

lambda. But the weight values are much smaller in magnitude when the lambda value is non-zero. This is due to the regularization parameter penalizing large values of lambda.

3.5 Comparing in terms of Error

The MSE for OLE training data is 8142.96 (with intercept)
The MSE for OLE test data is 11,408 (with intercept)
The best MSE using Ridge for the test data is 8773.
It is apparent that Ridge regression outperforms OLE with or without an intercept. Thus we can say that Ridge ression is better than OLE for this problem.

3.6 Optimal value of lambda

The optimal value of lambda was found to be 0.06. This is the ridge parameter value at which the lowest MSE for the test data was recorded. This value of lambda is perfect for regularization without underfitting.

4. Problem 4: Using Gradient Descent for Ridge Regression

4.1 Problem Requirement

Plot the errors on train and test data obtained by using gradient descent based learning by varying the regularization parameter λ. Compare with the results obtained in Problem 3.

4.2 Analysis

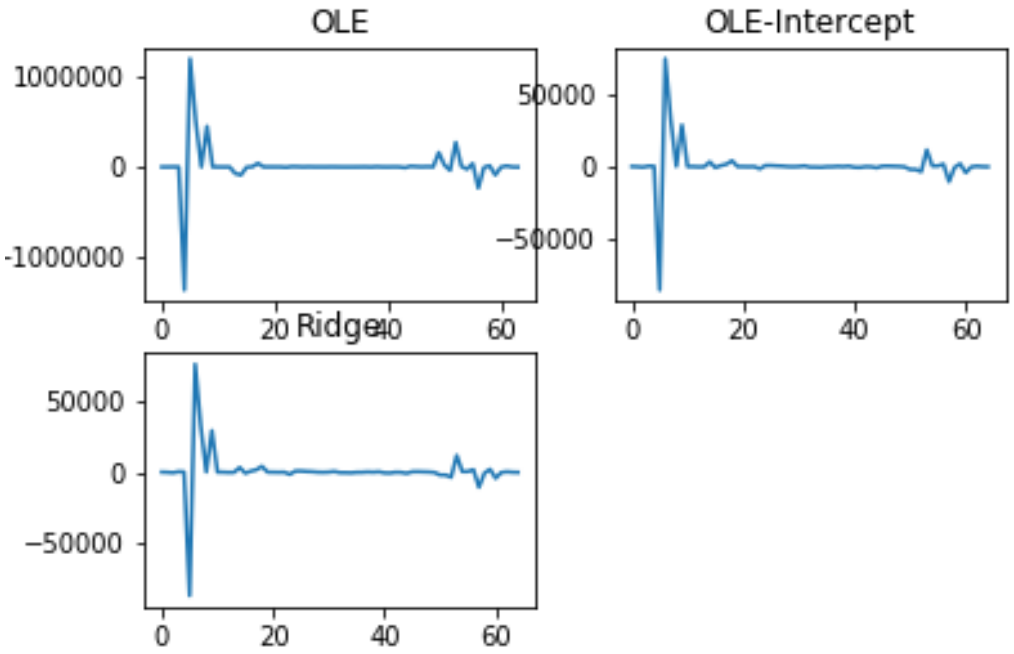

Figure 4: Regression weights

4.3 Varying Parameter value Errors

The graph for error varying with values of lambda that we get from gradient descent is very similar to the one we obtained in Problem 3. We notice that the graph, especially for the test data, is not as smooth as the graph from problem 3. This can be explained by outliers.
There are some situations in which gradient descent is favourable, even though it is more computationally expensive, especially with huge matrices which take a lot of computation to invert.

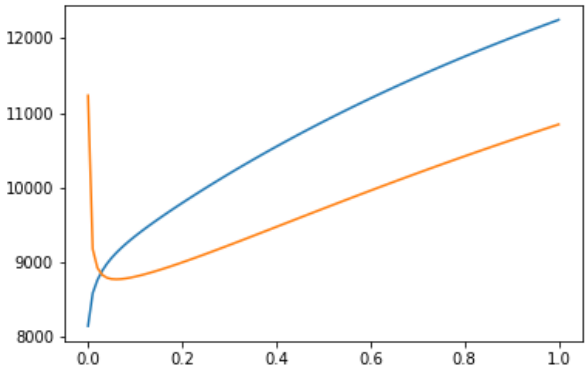

Figure 5: Gradient Descent

5. Problem 5: Non-Linear Regression

5.1 Problem Requirement

Using the λ = 0 and the optimal value of λ found in Problem 3, train ridge regression weights using the non-linear mapping of the data. Vary p from 0 to 6. Note that p = 0 means using a horizontal line as the regression line, p = 1 is the same as linear ridge regression. Compute the errors on train and test data. Compare the results for both values of λ. What is the optimal value of p in terms of test error in each setting? Plot the curve for the optimal value of p for both values of λ and compare.

5.2 Plot the Error with varying values of p

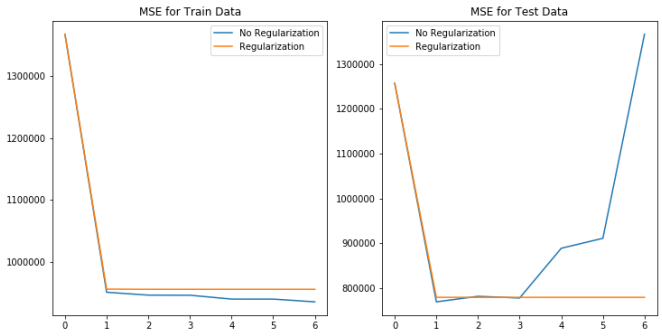

Figure 6: Error with different orders(p)

5.3 Optimal value of p

The optimal value of p with no regularization is 1.
The optimal value of p with regularization is 3.

6. Problem 6: Interpreting the results

6.1 Comparison of various algorithms

A list of all the algorithms implemented and their errors on the training and test data set are shown on page 9 in the form of a table.

The main focus in this problem is to minimize the error of the model, which gives us no reason to select OLE without an intercept, since it gives us an extremely high error in both training and testing.

Linear regression with intercept seems to give us the best training error, but it generalizes worse than the Ridge regression models that we have built, so we will not be considering OLE with intercept as well.

The best accuracy is obtained from both the Ridge regression models. However, in cases where the size of the data is very high, matrix inversion for classical Ridge regression may not be a feasable option. In this case, gradient descent is recommended as it provides an acceptable approximation to the optimal result (optimal if convex search space).

6.2 Conclusion

In conclusion, we infer from our results that the Ridge regression methods give us the highest accuracy out of all the models as they as less likely to overfit due to the regularization parameter. However, if we take computaional resources into consideration, classical Ridge regression would take fewer resources to compute for smaller data sets, and gradient descent would take fewer resources to compute for larger data sets where matrix inversion is very difficult.

Lambda	Train MSE	Test MSE
0.0	8142.966021125447	11408.739019091578
0.09	9299.279116506488	8795.213603534079
0.1	9350.255572943433	8808.485973617524
0.11	9399.280143429785	8823.50125897569
0.12	9446.767473272488	8839.883946804393
0.22	9876.482131873696	9043.772091894893
0.23	9916.654781327486	9066.475765487741
0.24	9956.454458376811	9089.430941588664
0.25	9995.895921521425	9112.611210491597
0.29	10150.297538311002	9207.137605565504
0.3	10188.098097910097	9231.125567576057
0.31	10225.591572974274	9255.223432579885
0.32	10262.783152089376	9279.417090874487
0.44	10686.983983805056	9573.280672265382
0.45	10720.59429256282	9597.813192199159
0.46	10753.951082999805	9622.322956166714
0.65	11343.201210115223	10078.859962446828
0.71	11513.367576635033	10217.913577392494
0.72	11541.058113320025	10240.808266021746
0.73	11568.562887895538	10263.620710607916
0.74	11595.884015383926	10286.35031658769
0.92	12058.50099336166	10680.932231974193
0.93	12082.697093598973	10702.03629017652
0.99	12224.83659328113	10826.861790458202
1.0	12248.034308841658	10847.367854255936

Algorithm	MSE Train	MSE Test
OLE without intercept	72219.7833	333673.0048
OLE with intercept	8142.9960	11408.7390
Ridge Regression	8148.65	8773.32
Ridge Regression with gradient descent	8148,65	8773.32
Non-linear regression without regularization	77911.65	95606.32
Non-linear regression with reularization	82733.76	95606.32