PA 2 - Classification and Regression

CSE 574: Introduction to Machine Learning

Team 7:

Firnaz Luztian Adiansyah - firnazlu@buffalo.edu

Sumedha Salil Nashte - sumedhas@buffalo.edu

Anushree Naveen Gupta - agupta38@buffalo.edu

UNIVERSITY AT BUFFALO

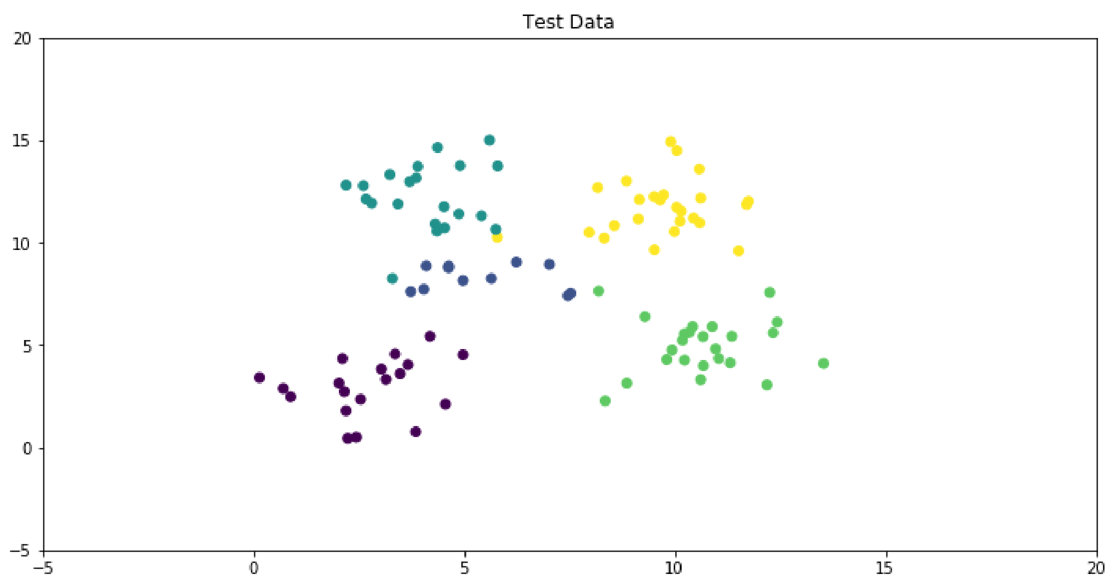
Problem 1: Experiment with Gaussian Discriminators

LDA and QDA follow similar approach except for one difference - for LDA, covariance matrix is calculated for all classes together. Whereas for QDA, we compute the covariance matrix for each of the classes. The mean is calculated for each of the possible output class separately for both LDA and QDA. Due to this difference LDA gives a linear classification and QDA gives a quadratic classification. This can be visibly seen in the plots below where for LDA we can see linear boundaries and for QDA we can see boundaries with curves.

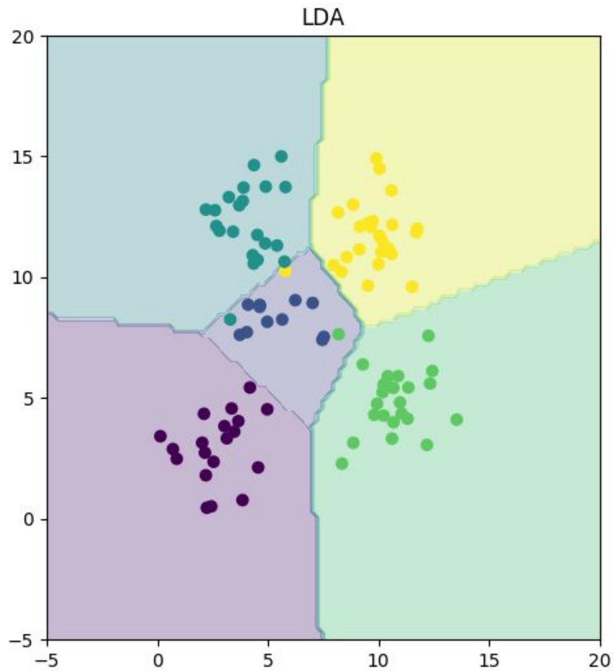
For the given data set we get the following accuracy for LDA and QDA:

- Accuracy for LDA : **97%**
- Accuracy for QDA: **96%**

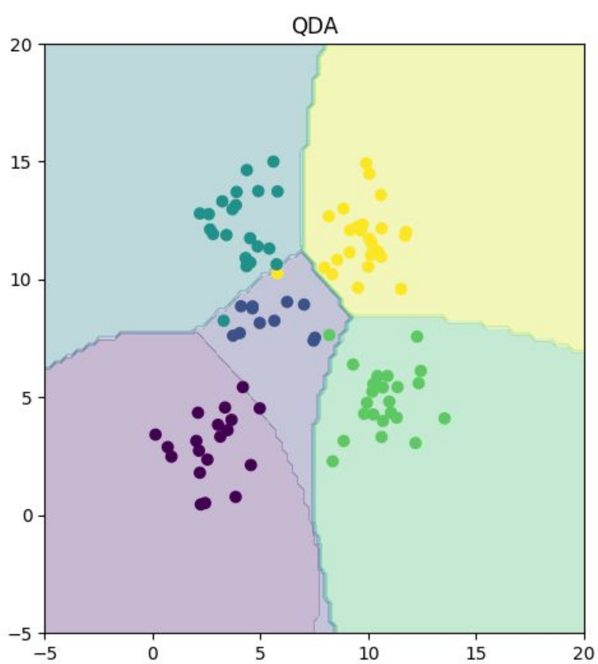
Test Data



Classification using LDA



Classification using QDA



Problem 2: Experiment with Linear Regression

We performed Linear Regression with and without intercept on test as well as training data. Below is the summary of the MSE values collected for all the different cases.

Data Set	MSE without intercept	MSE with intercept
Training data	19099.4468446	2187.16029493
Test Data	106775.361558	3707.84018132

Conclusion:

Mean Squared Error (MSE) is basically the difference between the estimator and what is estimated. The the lower value of MSE indicates a better prediction and can be said to be a better fit.

In our experiments, we observed that MSE values for training data are smaller compared to those for test data for every case. Moreover, calculating the MSE for both Training and Test data with intercept allow us to get a significant amount of decrease in error as compared to calculating the MSE without intercept. As stated above it's desired to get a lower MSE, thus we can conclude that calculating MSE values with intercept gives us a better estimation.

Problem 3: Experiment with Ridge Regression

We vary the lambda value from 0 to 1 in steps of 0.01 and have calculated the errors for each for them:

Lambda: 0.0 Train data: 2187.16029493 Test data: 3707.84018132	Lambda: 0.35 Train data: 2786.02671854 Test data: 3039.54529713	Lambda: 0.7 Train data: 3084.94542842 Test data: 3313.354623
Lambda: 0.01 Train data: 2306.83221793 Test data: 2982.44611971	Lambda: 0.36 Train data: 2795.70356824 Test data: 3047.49335111	Lambda: 0.71 Train data: 3092.43344001 Test data: 3320.82191265
Lambda: 0.02 Train data: 2354.07134393 Test data: 2900.97358708	Lambda: 0.37 Train data: 2805.30482034 Test data: 3055.45419817	Lambda: 0.72 Train data: 3099.87098085 Test data: 3328.26268646
Lambda: 0.03 Train data: 2386.7801631 Test data: 2870.94158888	Lambda: 0.38 Train data: 2814.83139806 Test data: 3063.42491285	Lambda: 0.73 Train data: 3107.25862691 Test data: 3335.67673095
Lambda: 0.04 Train data: 2412.119043 Test data: 2858.00040957	Lambda: 0.39 Train data: 2824.28419133 Test data: 3071.40277169	Lambda: 0.74 Train data: 3114.59694628 Test data: 3343.06385289
Lambda: 0.05 Train data: 2433.1744367 Test data: 2852.66573517	Lambda: 0.4 Train data: 2833.66406312 Test data: 3079.38523776	Lambda: 0.75 Train data: 3121.88649919 Test data: 3350.42387813
Lambda: 0.06 Train data: 2451.52849064 Test data: 2851.33021344	Lambda: 0.41 Train data: 2842.97185452 Test data: 3087.36994673	Lambda: 0.76 Train data: 3129.12783807 Test data: 3357.75665047
Lambda: 0.07 Train data: 2468.07755253 Test data: 2852.34999406	Lambda: 0.42 Train data: 2852.2083886 Test data: 3095.35469418	Lambda: 0.77 Train data: 3136.3215076 Test data: 3365.0620307
Lambda: 0.08 Train data: 2483.36564653 Test data: 2854.87973918	Lambda: 0.43 Train data: 2861.3744735 Test data: 3103.33742413	Lambda: 0.78 Train data: 3143.46804472 Test data: 3372.33989556
Lambda: 0.09 Train data: 2497.74025857 Test data: 2858.44442115	Lambda: 0.44 Train data: 2870.47090474 Test data: 3111.31621849	Lambda: 0.79 Train data: 3150.56797875 Test data: 3379.59013686
Lambda: 0.1 Train data: 2511.43228199 Test data: 2862.75794143	Lambda: 0.45 Train data: 2879.49846701 Test data: 3119.28928746	Lambda: 0.8 Train data: 3157.62183137 Test data: 3386.81266063
Lambda: 0.11 Train data: 2524.60003852 Test data: 2867.63790917	Lambda: 0.46 Train data: 2888.45793552 Test data: 3127.25496075	Lambda: 0.81 Train data: 3164.63011677 Test data: 3394.00738631
Lambda: 0.12 Train data: 2537.35489985 Test data: 2872.96228271	Lambda: 0.47 Train data: 2897.35007697 Test data: 3135.21167941	Lambda: 0.82 Train data: 3171.59334168 Test data: 3401.17424594
Lambda: 0.13 Train data: 2549.77688678 Test data: 2878.64586939	Lambda: 0.48 Train data: 2906.17565032 Test data: 3143.15798839	Lambda: 0.83 Train data: 3178.51200544 Test data: 3408.31318353
Lambda: 0.14 Train data: 2561.92452773 Test data: 2884.62691417	Lambda: 0.49 Train data: 2914.93540723 Test data: 3151.09252966	Lambda: 0.84 Train data: 3185.38660008 Test data: 3415.42415428
Lambda: 0.15 Train data: 2573.84128774 Test data: 2890.85910969	Lambda: 0.5 Train data: 2923.63009243 Test data: 3159.01403582	Lambda: 0.85 Train data: 3192.21761044 Test data: 3422.50712403
Lambda: 0.16 Train data: 2585.55987497 Test data: 2897.30665895	Lambda: 0.51 Train data: 2932.26044392 Test data: 3166.92132421	Lambda: 0.86 Train data: 3199.0055142 Test data: 3429.56206859
Lambda: 0.17 Train data: 2597.10519217 Test data: 2903.94112629	Lambda: 0.52 Train data: 2940.82719309 Test data: 3174.81329145	Lambda: 0.87 Train data: 3205.75078202 Test data: 3436.58897321
Lambda: 0.18 Train data: 2608.49640025 Test data: 2910.73937213	Lambda: 0.53 Train data: 2949.33106473 Test data: 3182.68890838	Lambda: 0.88 Train data: 3212.45387757 Test data: 3443.58783202
Lambda: 0.19 Train data: 2619.74838623 Test data: 2917.68216413	Lambda: 0.54 Train data: 2957.77277699 Test data: 3190.54721533	Lambda: 0.89 Train data: 3219.11525768 Test data: 3450.55864755
Lambda: 0.2 Train data: 2630.8728232 Test data: 2924.75322165	Lambda: 0.55 Train data: 2966.15304137 Test data: 3198.38731777	Lambda: 0.9 Train data: 3225.73537241 Test data: 3457.50143021
Lambda: 0.21 Train data: 2641.87894616 Test data: 2931.93854417	Lambda: 0.56 Train data: 2974.47256259 Test data: 3206.20838225	Lambda: 0.91 Train data: 3232.31466512 Test data: 3464.41619786
Lambda: 0.22 Train data: 2652.77412633 Test data: 2939.22592987	Lambda: 0.57 Train data: 2982.73203851 Test data: 3214.00963255	Lambda: 0.92 Train data: 3238.8535726 Test data: 3471.30297539
Lambda: 0.23 Train data: 2663.56430077 Test data: 2946.60462378	Lambda: 0.58 Train data: 2990.93215999 Test data: 3221.79034621	Lambda: 0.93 Train data: 3245.35252514 Test data: 3478.16179431
Lambda: 0.24 Train data: 2674.25429667 Test data: 2954.06505602	Lambda: 0.59 Train data: 2999.07361078 Test data: 3229.5498512	Lambda: 0.94 Train data: 3251.81194665 Test data: 3484.99269234
Lambda: 0.25 Train data: 2684.84807809 Test data: 2961.59864341	Lambda: 0.6 Train data: 3007.15706742 Test data: 3237.28752288	Lambda: 0.95 Train data: 3258.23225474 Test data: 3491.79571308
Lambda: 0.26 Train data: 2695.34893502 Test data: 2969.19763677	Lambda: 0.61 Train data: 3015.1831991 Test data: 3245.00278108	Lambda: 0.96 Train data: 3264.61386081 Test data: 3498.57090566
Lambda: 0.27 Train data: 2705.75962912 Test data: 2976.85500119	Lambda: 0.62 Train data: 3023.15266757 Test data: 3252.69508746	Lambda: 0.97 Train data: 3270.95717015 Test data: 3505.3183244
Lambda: 0.28 Train data: 2716.0825067 Test data: 2984.56432079	Lambda: 0.63 Train data: 3031.06612707 Test data: 3260.36394297	Lambda: 0.98 Train data: 3277.26258207 Test data: 3512.03802854
Lambda: 0.29 Train data: 2726.31958674 Test data: 2992.31972181	Lambda: 0.64 Train data: 3038.92422416 Test data: 3268.00888553	Lambda: 0.99 Train data: 3283.53048993 Test data: 3518.7300819
Lambda: 0.3 Train data: 2736.4726296 Test data: 3000.11580946	Lambda: 0.65 Train data: 3046.72759776 Test data: 3275.6294878	Lambda: 1.0 Train data: 3289.7612813 Test data: 3525.39455263
Lambda: 0.31 Train data: 2746.54319109 Test data: 3007.94761559	Lambda: 0.66 Train data: 3054.47687898 Test data: 3283.22535516	
Lambda: 0.32 Train data: 2756.53266482 Test data: 3015.81055453	Lambda: 0.67 Train data: 3062.17269114 Test data: 3290.79612376	
Lambda: 0.33 Train data: 2766.44231574 Test data: 3023.70038563	Lambda: 0.68 Train data: 3069.81564971 Test data: 3298.34145873	
Lambda: 0.34 Train data: 2776.27330654 Test data: 3031.61318093	Lambda: 0.69 Train data: 3077.40636224 Test data: 3305.86105245	

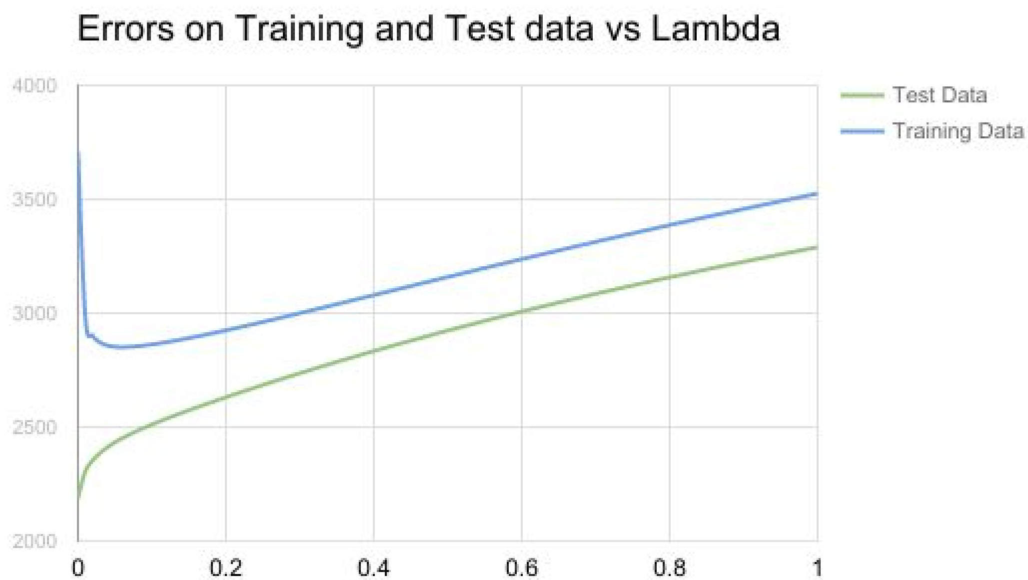
Below are the MSE values for training and test data using ridge regression for optimal λ = 0.06

MSE Values	MSE with intercept
Training data	2451.52849064
Test Data	2851.33021344

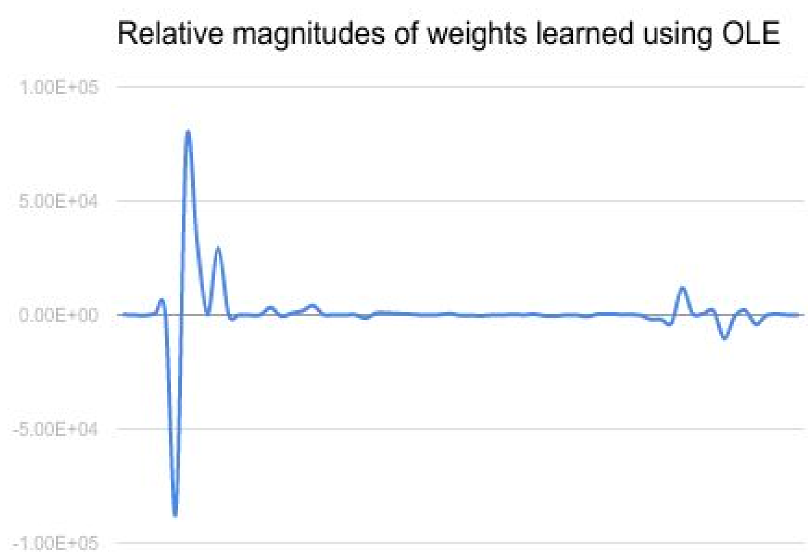
Conclusion:

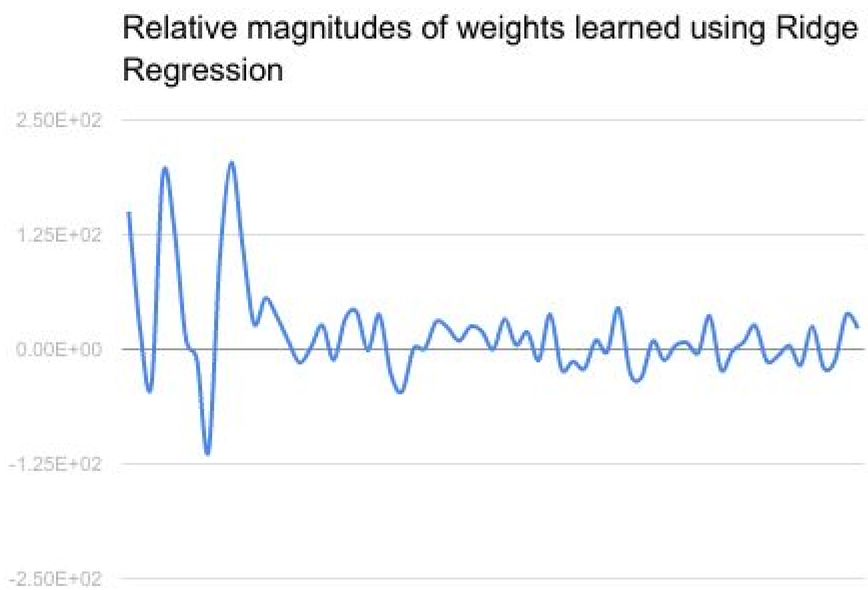
On observation of all the values we found that for testing data the MSE value slowly decreases till λ = 0.06 and then it starts to increase after it hits 0.06. Therefore, we can conclude that λ = 0.06 is optimal.

Plot for errors on train and test data for different values of λ



Comparison of the relative magnitudes of weights learnt using OLE and weights learnt using ridge regression





MSE Values for both the train and test data using intercept with both the approaches are as below:

Data Sets	MSE Values with intercept	
	OLE	Ridge Regression
Training data	2187.16029493	2451.52849064
Test Data	3707.84018132	2851.33021344

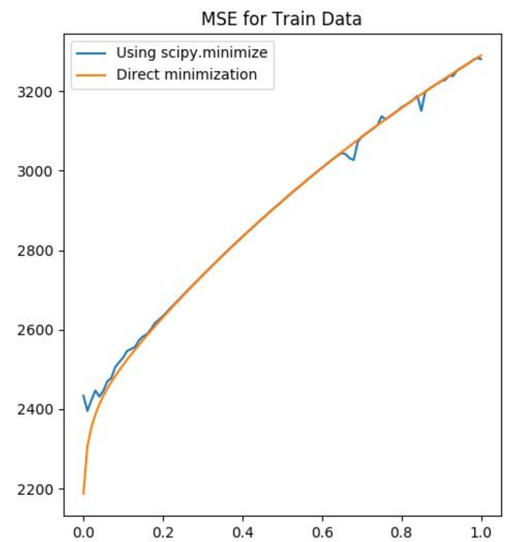
Conclusion:

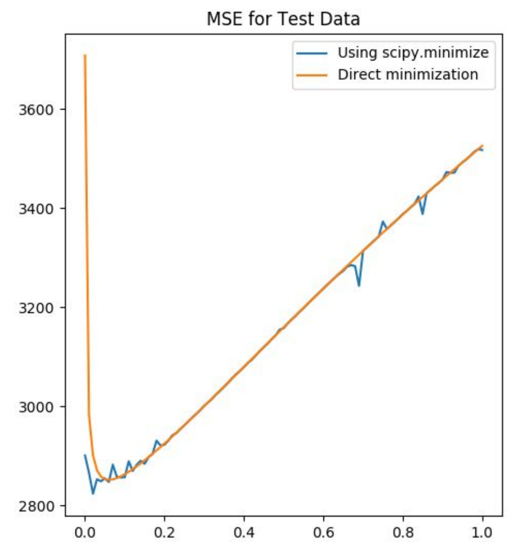
Observing the MSE values we can state that even though we get a lower MSE using OLE for train data, Ridge regression works better on the test data. As the difference between MSE values for training data are comparatively quite less when compared to those for test data we can say that Ridge regression gives us a better estimation.

Even in general practises, Ridge has two main benefits over OLE - adding a penalty term which reduces overfitting as well as guarantees that we can find a solution.

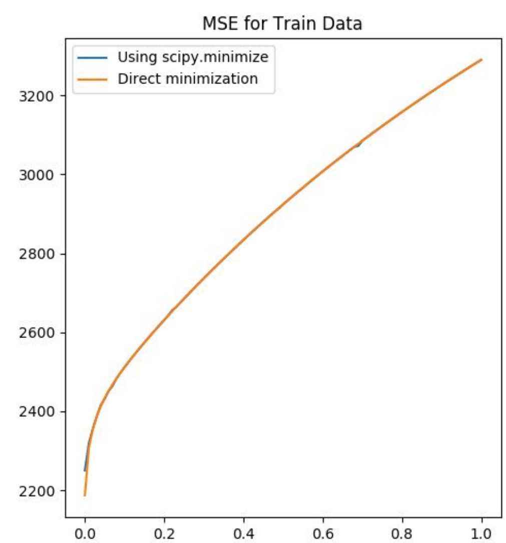
Problem 4: Using Gradient Descent for Ridge Regression Learning

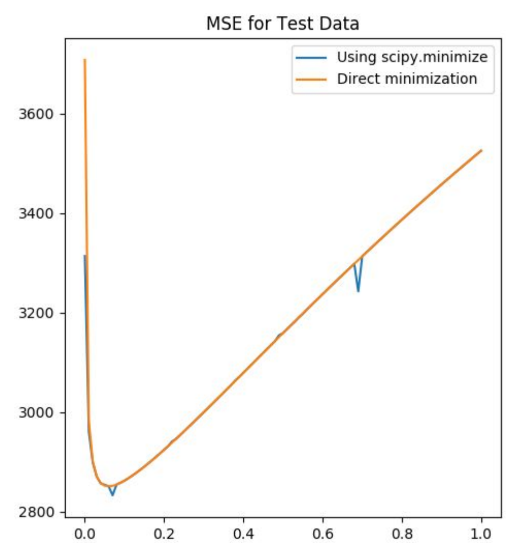
Maximum iteration: 20





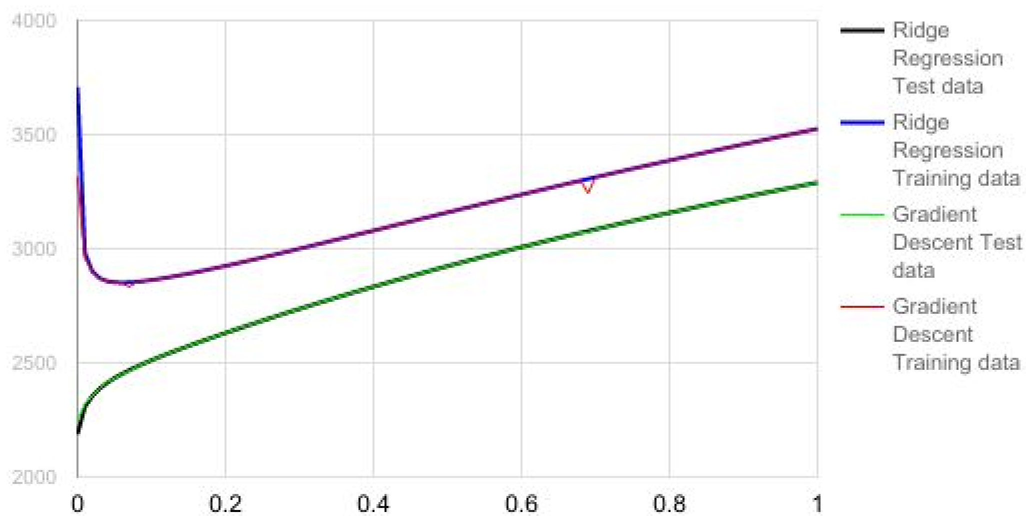
Maximum iteration: 100





Plot for errors on train and test data obtained by using the gradient descent based learning by varying the regularization parameter λ

Compare the errors for Ridge Regression and Gradient Descent



Lambda = 0.06	**Training data**	**Test data**
Error using gradient descent based learning	2451.53068977	2851.32857305

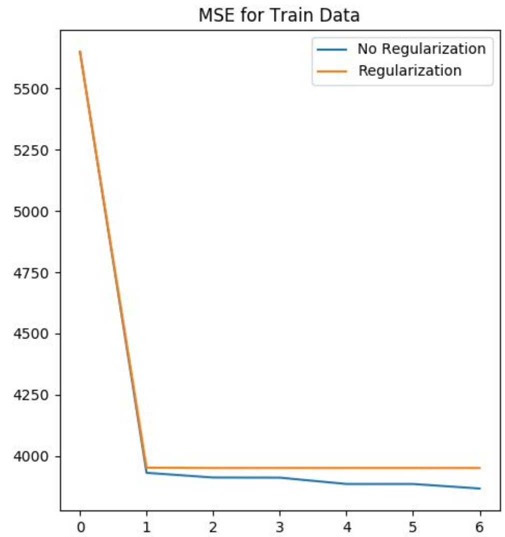
Conclusion:

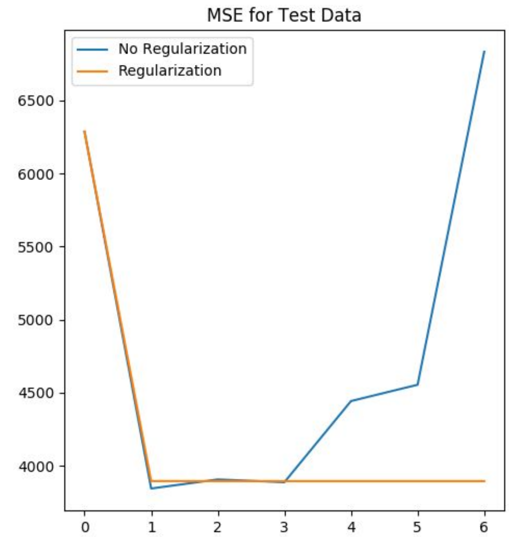
The MSE values we obtained in Problem 4 using Gradient Descent are almost similar to those obtained in Problem 3 for Ridge Regression. The only difference which can be seen is that for Gradient Descent the plot doesn't seem to be smooth. There are a few minor peaks and troughs which can seen.

Ridge regression through intercept is better than Ridge regression through Gradient Descent in terms of performance as the minimize function in Gradient Descent can take some time to converge. But for bigger datasets Ridge regression through Gradient Descent is better as each step is easy to compute and doesn't involve any expensive computations.

Problem 5: Non-linear Regression

Ridge regression weights using non-linear mapping





Errors on train data and test data for both λ = 0 and λ Optimal:

P	Training Data		Test Data	
	λ = 0	λ = 0.06	λ = 0	λ = 0.06
0	5650.7105389	5650.71190703	6286.40479168	6286.88196694
1	3930.91540732	3951.83912356	**3845.03473017**	3895.85646447
2	3911.8396712	3950.68731238	3907.12809911	3895.58405594
3	3911.18866493	3950.68253152	3887.97553824	3895.58271592
4	3885.47306811	3950.6823368	4443.32789181	**3895.58266828**
5	3885.4071574	3950.68233518	4554.83037743	3895.5826687
6	**3866.88344945**	**3950.68233514**	6833.45914872	3895.58266872

Therefore, the optimal values for p are:

Training Data		Test Data	
With regularization	Without regularization	With regularization	Without regularization
6	6	1	4

Problem 6: Interpreting Results

Summarization of MSE values using the various approaches for training and testing data sets.

P	Classifiers	Training data	Test data
2	OLE without intercept	19099.4468446	106775.361558
2	OLE with intercept	**2187.16029493**	3707.84018132
3	Ridge Regression (Optimal λ = 0.06)	2451.52849064	2851.33021344
4	Ridge Regression with Gradient Descent	2451.53068977	**2851.32857305**
5	Optimal without regularization	3866.88344945	3845.03473017
5	Optimal with regularization	3950.68233514	3895.58266828

Conclusion:

As per our experiment values of Mean Squared Error (MSE), the best classifier for Training data is Linear Regression(OLE) with Intercept whereas that for Test data is Ridge Regression with Gradient Descent. Considering a general case we can say that Ridge Regression will be the best approach for classifications. Thus for small data sets MSE is a good metric for measurement and analysis.

When it comes to big datasets, we even need to consider the running time for the process.

In those cases, RIdge regression can be infeasible due to expensive computation it involves. There we can use Gradient Descent which provides faster processing with acceptable range of errors.