DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY AT BUFFALO

CSE 574 (Machine Learning)
Programming Assignment 1
"Classification and Regression"

Group 141

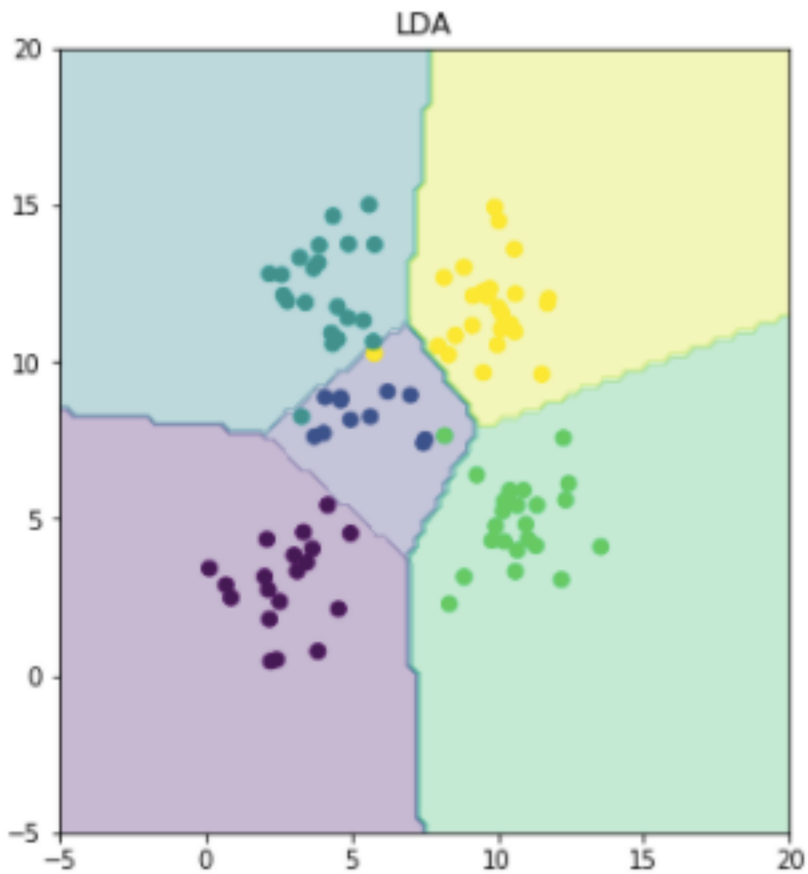
Shivam Sahu(ssahu3@buffalo.edu) Parush Garg(parushga@buffalo.edu)

PROBLEM 1 - EXPERIMENT WITH GAUSSIAN DISCRIMINATORS

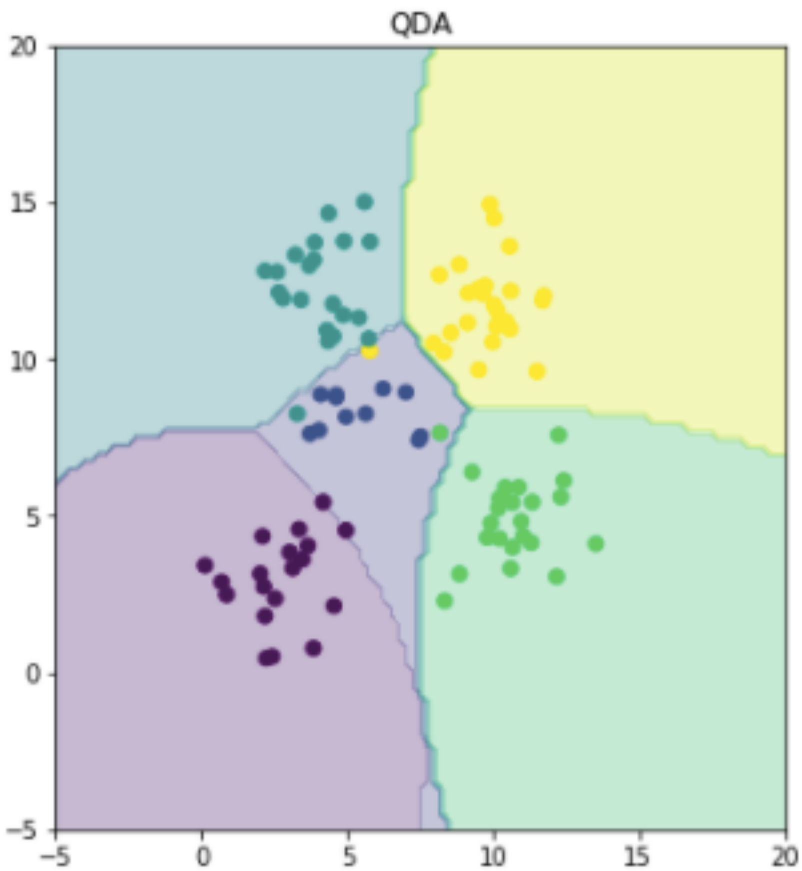
LDA accuracy = 97%
QDA accuracy = 95%

For LDA, we assign the same covariance matrix to all the classes. So the discriminating boundaries will show up as straight lines.
For QDA, we assign the different covariance matrix depending on classes. So, the discriminating boundaries are curved lines and the pace of pdf changes is not the same for all of them.

Discriminating boundary for LDA:



Discriminating boundary for QDA:



PROBLEM 2 - EXPERIMENT WITH LINEAR REGRESSION

```
MSE for test data without intercept [ 106775.36145119]
MSE for test data with intercept [ 3707.84018096]
MSE for train data without intercept [ 19099.44684457]
MSE for train data with intercept [ 2187.16029493]
```

For test data, using intercept showed the bigger amount of decrease in error. More importantly, it is very clear as using intercept, error is decreased by 96.5% i.e. from 106775.36145 to 3707.8401 which is a major decrement in error. Therefore, using intercept while finding the MSE for any particular is anytime better than not using intercept.

For train data, using intercept showed the bigger but lesser than train data, amount of decrease in error. More importantly, it is very clear as using intercept, error is decreased by 88.5% i.e. from 19099.4468 to 2187.1602 which is a major decrement in error.

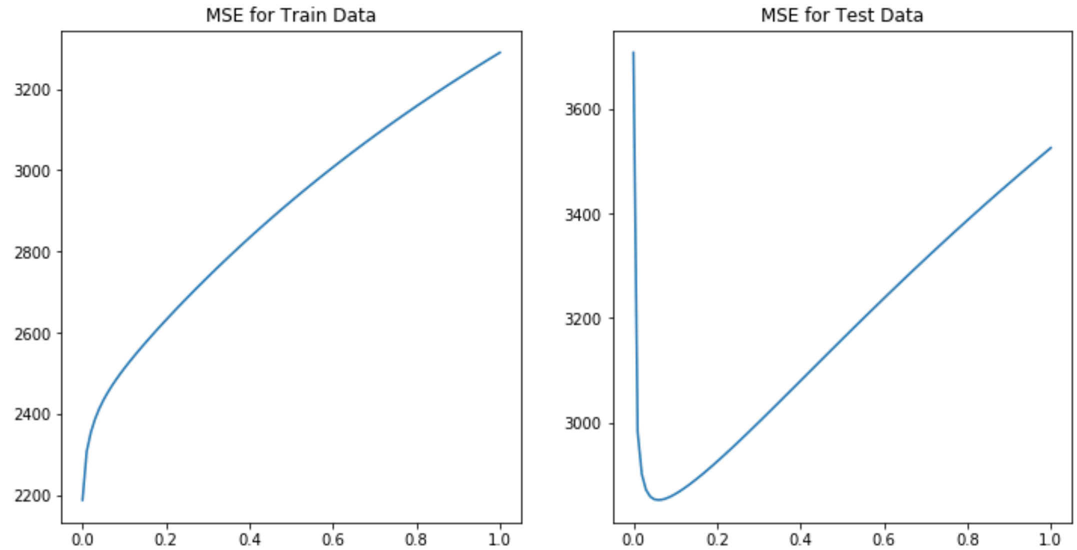
PROBLEM 3 - EXPERIMENT WITH RIDGE REGRESSION

MSE values for train and test data with Lambda varying from 0 to 1 in increment of 0.01

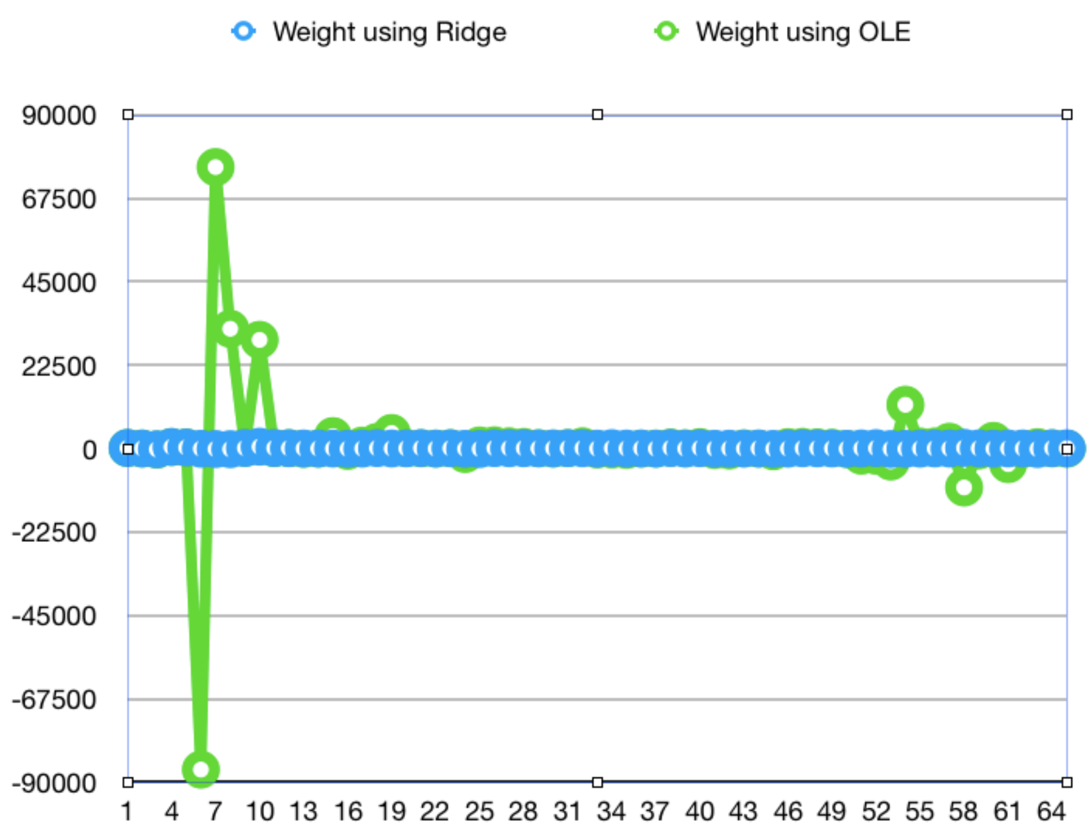
Lambda	Train	Test	Lambda	Train	Test	Lambda	Train	Test
0	[2187.16029493]	[3707.84018132]	0.38	[2814.83139806]	[3063.42491285]	0.76	[3129.12783807]	[3357.75665047]
0.01	[2306.83221793]	[2982.44611971]	0.39	[2824.28419133]	[3071.40277169]	0.77	[3136.3215076]	[3365.0620307]
0.02	[2354.07134393]	[2900.97358708]	0.4	[2833.66406312]	[3079.38523776]	0.78	[3143.46804472]	[3372.33989556]
0.03	[2386.7801631]	[2870.94158888]	0.41	[2842.97185452]	[3087.36994673]	0.79	[3150.56797875]	[3379.59013686]
0.04	[2412.119043]	[2858.00040957]	0.42	[2852.2083886]	[3095.35469418]	0.8	[3157.62183137]	[3386.81266063]
0.05	[2433.1744367]	[2852.66573517]	0.43	[2861.3744735]	[3103.33742413]	0.81	[3164.63011677]	[3394.00738631]
0.06	[2451.52849064]	[2851.33021344]	0.44	[2870.47090474]	[3111.31621849]	0.82	[3171.59334168]	[3401.17424594]
0.07	[2468.07755253]	[2852.34999406]	0.45	[2879.49846701]	[3119.28928746]	0.83	[3178.51200544]	[3408.31318353]
0.08	[2483.36564653]	[2854.87973918]	0.46	[2888.45793552]	[3127.25496075]	0.84	[3185.38660008]	[3415.42415428]
0.09	[2497.74025857]	[2858.44442115]	0.47	[2897.35007697]	[3135.21167941]	0.85	[3192.21761044]	[3422.50712403]
0.1	[2511.43228199]	[2862.75794143]	0.48	[2906.17565032]	[3143.15798839]	0.86	[3199.0055142]	[3429.56206859]
0.11	[2524.60003852]	[2867.63790917]	0.49	[2914.93540723]	[3151.09252966]	0.87	[3205.75078202]	[3436.58897321]
0.12	[2537.35489985]	[2872.96228271]	0.5	[2923.63009243]	[3159.01403582]	0.88	[3212.45387757]	[3443.58783202]
0.13	[2549.77688678]	[2878.64586939]	0.51	[2932.26044392]	[3166.92132421]	0.89	[3219.11525768]	[3450.55864755]
0.14	[2561.92452773]	[2884.62691417]	0.52	[2940.82719309]	[3174.81329145]	0.9	[3225.73537241]	[3457.50143021]
0.15	[2573.84128774]	[2890.85910969]	0.53	[2949.33106473]	[3182.68890838]	0.91	[3232.31466512]	[3464.41619786]
0.16	[2585.55987497]	[2897.30665895]	0.54	[2957.77277699]	[3190.54721533]	0.92	[3238.8535726]	[3471.30297539]
0.17	[2597.10519217]	[2903.94112629]	0.55	[2966.15304137]	[3198.38731777]	0.93	[3245.35252514]	[3478.16179431]
0.18	[2608.49640025]	[2910.73937213]	0.56	[2974.47256259]	[3206.20838225]	0.94	[3251.81194665]	[3484.99269234]
0.19	[2619.74838623]	[2917.68216413]	0.57	[2982.73203851]	[3214.00963255]	0.95	[3258.23225474]	[3491.79571308]
0.2	[2630.8728232]	[2924.75322165]	0.58	[2990.93215999]	[3221.79034621]	0.96	[3264.61386081]	[3498.57090566]
0.21	[2641.87894616]	[2931.93854417]	0.59	[2999.07361078]	[3229.5498512]	0.97	[3270.95717015]	[3505.3183244]
0.22	[2652.77412633]	[2939.22592987]	0.6	[3007.15706742]	[3237.28752288]	0.98	[3277.26258207]	[3512.03802854]
0.23	[2663.56430077]	[2946.60462378]	0.61	[3015.1831991]	[3245.00278108]	0.99	[3283.53048993]	[3518.7300819]
0.24	[2674.25429667]	[2954.06505602]	0.62	[3023.15266757]	[3252.69508746]	1	[3289.7612813]	[3525.39455263]
0.25	[2684.84807809]	[2961.59864341]	0.63	[3031.06612707]	[3260.36394297]			
0.26	[2695.34893502]	[2969.19763677]	0.64	[3038.92422416]	[3268.00888553]			
0.27	[2705.75962912]	[2976.85500119]	0.65	[3046.72759776]	[3275.6294878]			
0.28	[2716.0825067]	[2984.56432079]	0.66	[3054.47687898]	[3283.22535516]			
0.29	[2726.31958674]	[2992.31972181]	0.67	[3062.17269114]	[3290.79612376]			
0.3	[2736.4726296]	[3000.11580946]	0.68	[3069.81564971]	[3298.34145873]			
0.31	[2746.54319109]	[3007.94761559]	0.69	[3077.40636224]	[3305.86105245]			
0.32	[2756.53266482]	[3015.81055453]	0.7	[3084.94542842]	[3313.354623]			
0.33	[2766.44231574]	[3023.70038563]	0.71	[3092.43344001]	[3320.82191265]			
0.34	[2776.27330654]	[3031.61318093]	0.72	[3099.87098085]	[3328.26268646]			
0.35	[2786.02671854]	[3039.54529713]	0.73	[3107.25862691]	[3335.67673095]			
0.36	[2795.70356824]	[3047.49335111]	0.74	[3114.59694628]	[3343.06385289]			
0.37	[2805.30482034]	[3055.45419817]	0.75	[3121.88649919]	[3350.42387813]			

In order to obtain the optimal value of Lambda, we look in the test data and MSE has to be minimum. Hence at 0.06, MSE for ridge test data is minimum. Lambda = 0.06

Plot the errors on train and test data for different values of Lambda.



Plot the errors on train and test data for different values of Lambda.



The weight values using OLE have higher values than the weight values using Ridge since Ridge uses regularization which helps in keeping weight values similar thereby not creating much difference in the values in case of Ridge expression.

The above graph has been plotted using the table provided which consists the weight values of Weight using Ridge and Weight using OLE.

<u>Weight using OLE and Ridge for Lambda = 0.06</u>

Table 1

Serial number	Weight using Ridge	Weight using OLE
1	150.4595981	1.48E+02
2	4.80776899	1.27E+00
3	-202.9061147	-2.93E+02
4	421.7194576	4.15E+02
5	279.4510729	2.72E+02
6	-52.29708233	-8.66E+04
7	-128.5941891	7.59E+04
8	-167.5005703	3.23E+04
9	145.740681	2.21E+02
10	496.3060412	2.93E+04
11	129.9484578	1.25E+02
12	88.30438076	9.44E+01
13	11.29067689	-9.39E+01
14	1.88532531	-3.37E+01
15	-2.58364157	3.35E+03
16	-66.89445481	-6.21E+02
17	-20.61939955	7.92E+02
18	113.3930145	1.77E+03
19	17.99086827	4.19E+03
20	52.50235963	1.19E+02
21	109.6876551	7.66E+01
22	-10.72779629	-1.52E+01
23	71.67974829	8.22E+01
24	-69.30906366	-1.46E+03
25	-124.0343729	8.27E+02
26	102.639818	8.69E+02
27	72.64220588	5.86E+02
28	79.24754013	4.27E+02
29	38.48319215	9.02E+01
30	32.98009446	-1.79E+01
31	92.09539122	1.42E+02
32	68.97936154	5.83E+02
33	-24.41700914	-2.34E+02
34	101.8538797	-2.56E+02
35	1.39122669	-3.85E+02
36	20.85757155	-3.34E+01
37	-29.65490134	-1.07E+01
38	130.4111599	2.57E+02
39	-16.75108796	6.00E+01
40	87.51340344	3.84E+02
41	-45.64238362	-4.04E+02
42	-30.92288499	-5.14E+02
43	-10.07139781	3.84E+01
44	31.13334896	-4.46E+01
45	-89.33525423	-7.30E+02
46	-22.73053674	3.77E+02
47	65.41116624	4.40E+02
48	55.11621318	3.09E+02
49	19.14925041	1.90E+02
50	-59.84315841	-1.10E+02
51	26.64350735	-1.92E+03
52	108.4050128	-1.92E+03
53	-137.6175697	-3.49E+03
54	-83.04383566	1.18E+04
55	-20.40214777	5.31E+02
56	24.9726362	5.43E+02
57	-0.92451093	1.82E+03
58	191.9130658	-1.05E+04
59	34.78309393	-5.17E+02
60	-43.90393505	2.06E+03
61	23.2002376	-4.20E+03
62	20.8504118	-1.40E+02
63	-117.853228	3.74E+02
64	75.30611309	5.15E+01
65	60.36839226	-4.64E+01

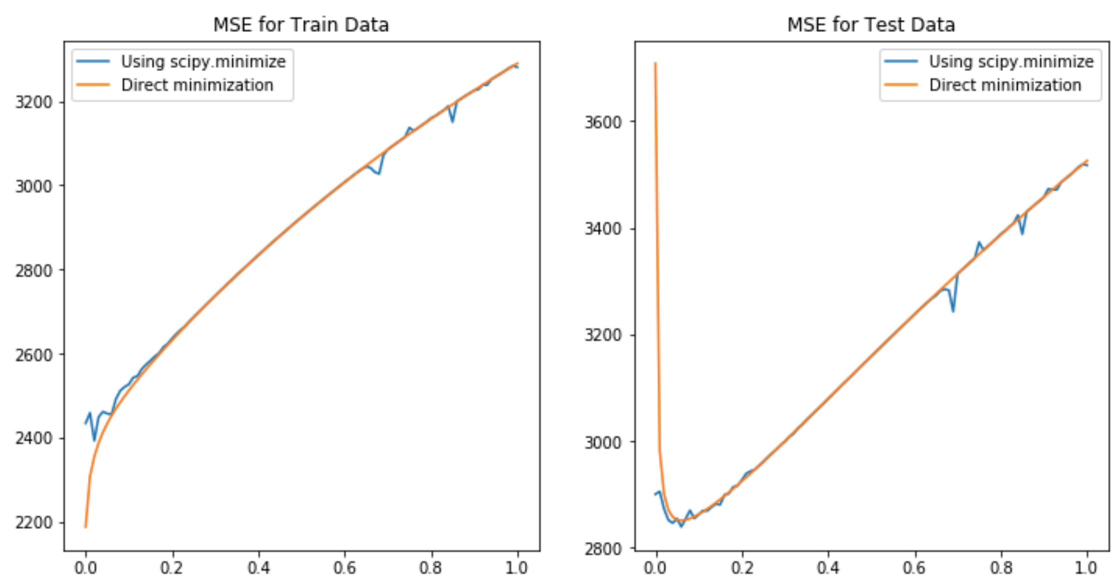
<u>Comparison of two approaches in terms of errors on train and test data and optimal value of lambda</u>

```
MSE for test data without intercept Linear Regression[ 106775.36145119]
MSE for test data with intercept Linear Regression[ 3707.84018096]
MSE for train data without intercept Linear Regression[ 19099.44684457]
MSE for train data with intercept Linear Regression[ 2187.16029493]
MSE for train data Ridge Regreesion [ 2451.52849064]
MSE for test data Ridge Regreesion [ 2851.33021344]
```

As it is visible from the table that MSE for ridge regression for train data = 2451.5284 and MSE for ridge regression for test data = 2851.3302. Moreover, MSE for OLE regression on test data is 3707.8401 and MSE for OLE regression on train data is 2187.1602. Clearly, MSE for Ridge regression is lower than OLE regression which makes Ridge regression better than OLE regression.

PROBLEM 4 – USING GRADIENT DESCENT FOR RIDGE REGRESSION LEARNING

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



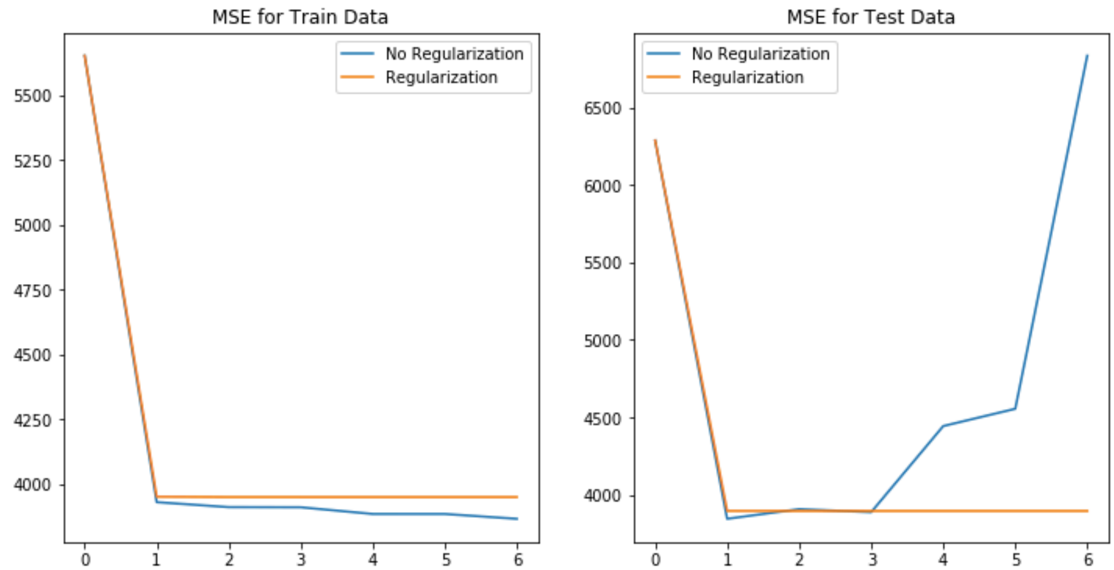
It is clearly visible that the graph for Ridge regression and Gradient descent for Ridge regression is nearly same. For Lambda value to be very low or very high, Gradient descent for Ridge regression graph is not as smooth as Ridge regression has. It has lot of ups and downs while for the middle values of lambda the graphs are identical. Moreover, the process to plot the graph for Gradient Descent function takes more time and is much slower than Ridge regression due to minimize () function and inverse as well. Moreover, the optimal value of Lambda is also similar to what we obtained in Problem 3 which is 0.06. Below attached is the MSE data obtained in Problem 4.

<u>MSE data obtained in Problem 4.</u>

	A	B	C	D
1	0	[2433.6773432]	[2900.54199346]	
2	0.01	[2458.31293877]	[2905.67666124]	
3	0.02	[2392.03040266]	[2873.0535521]	
4	0.03	[2448.62539812]	[2852.13981294]	
5	0.04	[2461.06596819]	[2846.50200165]	
6	0.05	[2456.5900618]	[2854.73097605]	
7	0.06	[2454.41484947]	[2839.4193483]	
8	0.07	[2491.24467966]	[2854.50699867]	
9	0.08	[2510.82848179]	[2870.21098082]	
10	0.09	[2520.01162464]	[2855.18987657]	
11	0.1	[2526.17557361]	[2862.34653486]	
12	0.11	[2542.14824896]	[2869.80954702]	
13	0.12	[2545.90638496]	[2869.35898237]	
14	0.13	[2562.63803225]	[2876.3643972]	
15	0.14	[2573.39886503]	[2882.3580064]	
16	0.15	[2582.42416651]	[2880.96733722]	
17	0.16	[2592.48403296]	[2899.48073317]	
18	0.17	[2600.22967365]	[2901.79339413]	
19	0.18	[2614.68452611]	[2914.06561383]	
20	0.19	[2622.89487766]	[2916.32901033]	
21	0.2	[2635.08570583]	[2926.83724041]	
22	0.21	[2646.00454496]	[2938.94331414]	
23	0.22	[2655.81159268]	[2943.58680204]	
24	0.23	[2664.10123691]	[2946.42098386]	
25	0.24	[2675.66011235]	[2954.00157072]	
26	0.25	[2686.12353776]	[2961.38200503]	
27	0.26	[2696.30156492]	[2969.40129793]	

PROBLEM 5 – NON-LINEAR REGRESSION

<u>Plot the errors on train and test data.</u>



Above picture is plotted for graph of MSE for train data and test data with regularization and without regularization as well. In no regularization, the value of Lambda used is 0.06 which we attained from the problem 3.

Graph 1 on the left side of the figure clearly shows that the error decreases as we increase the value of p i.e. we increase the degree of polynomial. At p = 1, both the blue lines (for lambda = 0.06) and orange line (regularization) have close to similar errors. After p = 1, regularization has a straight line along x axis and no regularization keeps on decreasing. This happens as the data set is train and higher order polynomial will fit the curve better than degree (x<=1). Therefore, there is a decrease in the error.

Graph 2 on the right side of the figure clearly shows that for no regularization, after p = 1, the error increases and grows high after the degree of polynomial starting from p = 3. However, for regularization, the graph is almost similar to what we got on the train data. It gets the minimum error at p = 1. This happens as we have over fitting with higher degree polynomial. Since we train the algorithm using train data hence it is more relying on train data. That is the reason of increment in error when run on test data.

Graph for Lambda = 0 without regularization with p varying 0 to 6.

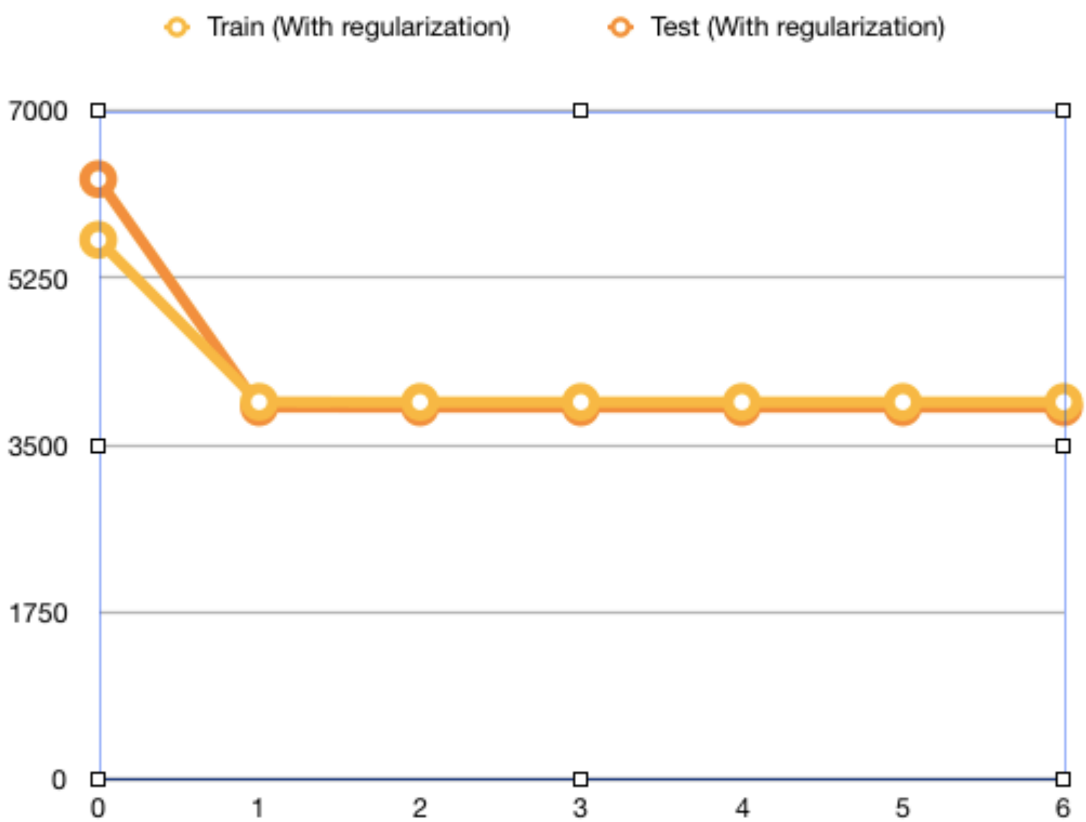
Table 1

p	Train (Without regularization)	Test (Without regularization)
0	5650.710539	6286.404792
1	3930.915407	3845.03473
2	3911.839671	3907.128099
3	3911.188665	3887.975538
4	3885.473068	4443.327892
5	3885.407157	4554.830377
6	3866.883449	6833.459149

Graph for Lambda = 0.06 with regularization with p varying 0 to 6.

Table 1

p	Train (With regularization)	Test (With regularization)
0	5650.711907	6286.881967
1	3951.839124	3895.856464
2	3950.687312	3895.584056
3	3950.682532	3895.582716
4	3950.682337	3895.582668
5	3950.682335	3895.582669
6	3950.682335	3895.582669



With regularization which is Lambda = 0.06, for test data:
The optimal value for p in case of test error is 4. It can be p = 2 or p = 3 since the line is a straight line thereafter.

With regularization which is Lambda = 0.06, for train data:
The optimal value for p in case of test error is 5 or 6. The lines are somewhat very much similar for both the dataset i.e. train and test data.

Without regularization, for test data:
The optimal value for p in case of test error is 1 since the error attained is minimum.
Without regularization, for train data:
The optimal value for p in case of test error is 5 since the error attained is minimum. The graphs are not similar since when the code is run for the test data, it tries to adjust in to the level of train data which creates lot of distortion in the graph.

PROBLEM 6 – INTERPRETING RESULTS

Type of Regression	**Train MSE**	**Test MSE**
MSE without intercept (Q2)	19099.4468	106775.36145
MSE with intercept (Q2)	2187.1602	3707.8401
MSE with Ridge regression (Q3)	2451.5284	2851.3302
MSE with Ridge regression using Gradient Descent (Q4)	2454.4148	2839.4193
MSE with Non-Linear regression without regularization (Q5)	3911.1886	3845.0347
MSE with Non-Linear regression with regularization (Q5)	3950.6823	3895.5826

In order to choose the best setting, metric used is error. If the error obtained is minimum, then that setting is the best solution. Using this approach, we have identified that for **training data**, **Linear regression** gives us the least error which is 2187.1602 hence provides the best solution. But for **testing data**, the least error is provided by **Ridge regression using Gradient Descent** which is 2839.4193 therefore providing the best solution which is better than Linear Regression.

Clearly, Linear regression without intercept can not be used since it produces the maximum error among all the settings. Moreover, Non-linear regression also gives very high error rate.
Moreover, we can consider the metric of run time as well. Clearly, due to matrix inversion in Ridge expression, it takes some time to plot the graph. This is happening in small data set and may increase the time duration in big data sets.

In a nutshell, it is clearly visible that **Ridge regression without gradient descent** and **Ridge regression using gradient descent** produce identical results as it can be seen from the table above. Hence, it is one of the suitable way to use Ridge regression. But runtime of Ridge regression is high which will be a problem for data sets which are big. Therefore, regularization helps in giving better results in these cases.