Department of Computer Science & Engineering

CSE 574 – MACHINE LEARNING

Classification and Regression

Programming Assignment – 1

Presented By:

Pratik **Agarwal** [50290570]
Sourav **Bihani** [50290969]
Tannu **Priya** [50290768]

Group - 23

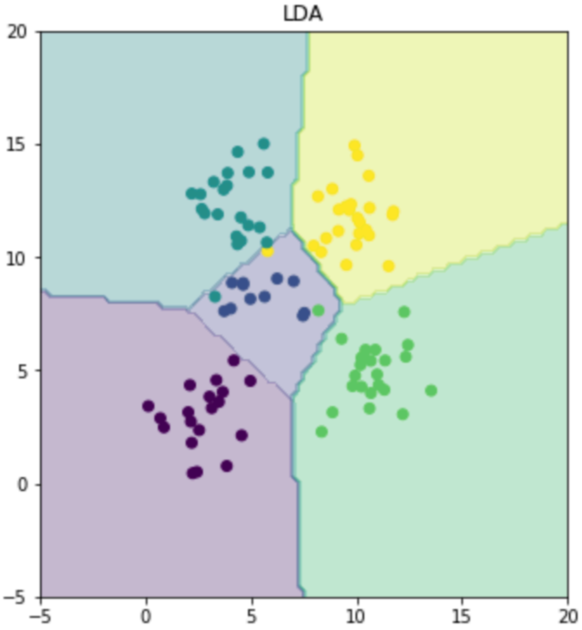
Table of Contents:

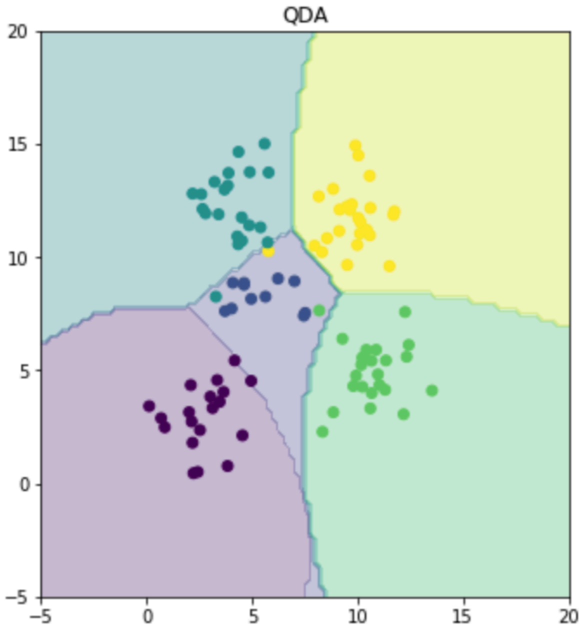
Problem 1: Experiment with Gaussian Discriminators

The following results were obtained:

Accuracy for Linear Discriminant Analysis (**LDA**) : **97%**
Accuracy for Quadratic Discriminant Analysis (**QDA**) : **96%**

The graphs obtained for LDA and QDA are:





For the above graphs, following can be inferred:

LDA	**QDA**
In LDA we can observe linear boundaries separating classes from each other.	In QDA the boundaries separating classes are non – linear/parabolic.
We achieved better accuracy in LDA i.e. **97%** because of the fact that LDA uses covariance of all the training data.	In QDA accuracy, the accuracy is **96%** because QDA uses covariance of every class for generating the predictions.
LDA can only learn lines.	QDA can learn curves as well.

Problem 2: Experiment with Linear Regression

The following results were obtained:

Category	MSE without Intercept	MSE with Intercept
Training Data	19099.4468446	2187.16029493
Testing Data	106775.36155405	3707.8401816

From the above data, following can be inferred:

MSE without Intercept	MSE with Intercept
Without intercept we are forcing the line to pass through the origin and hence data does not fit well with the actual data.	With Intercept the Linear regression line has more flexibility and hence is closely aligned with the actual data.
Improvement in error is only **88.55%** without intercept on a single data set.	There is a significant improvement of **96.53%** with intercept on a single data set.

Hence, we can conclude that **MSE with Intercept** has better accuracy.

Problem 3: Experiment with Ridge Regression

Ridge and Linear Regression are almost the same, apart from an additional parameter λ (*Regularization Parameter*) introduced in Ridge Regression. This regularization parameter is fed back to the system with a value that would adjust the weights in order to minimize the Mean Square Error (MSE).

To determine the best value of λ (*Regularization Parameter*), we use the cross-validation approach. We are trying to fit a model to our training dataset for a specific value of λ. Once the value of λ is determined, we can apply the model to the test dataset.

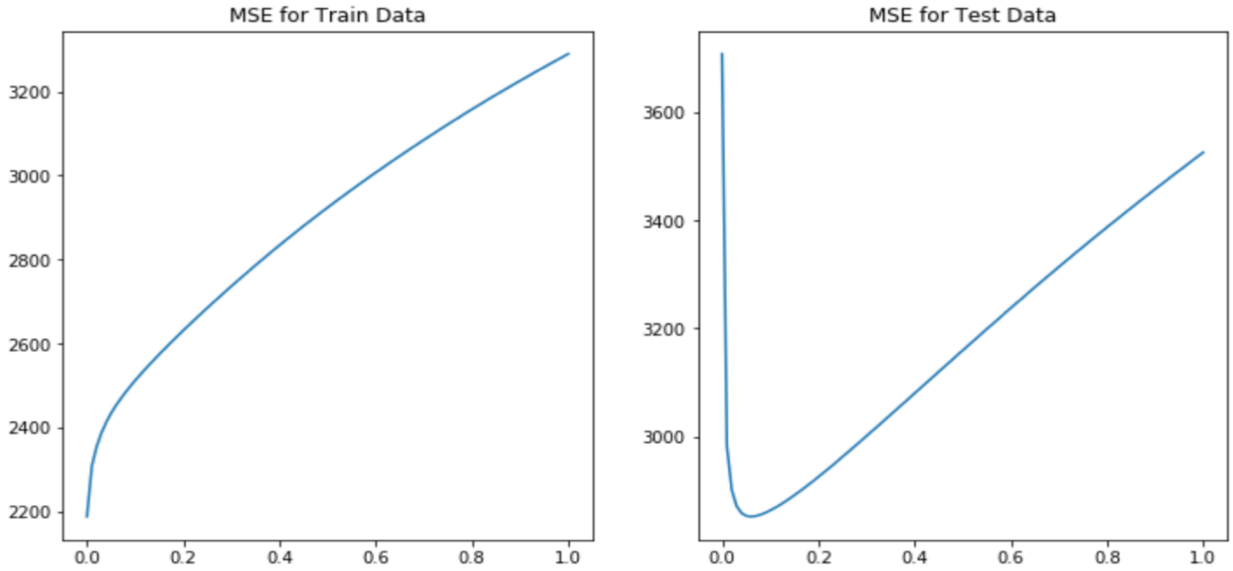
MSE for Training and Test Dataset for $0.0 \leq \lambda \leq 1.0$

Lambda	MSE Train	MSE Test
0	[2187.16029493]	[3707.84018171]
0.01	[2306.83221793]	[2982.44611971]
0.02	[2354.07134393]	[2900.97358708]
0.03	[2386.7801631]	[2870.94158888]
0.04	[2412.119043]	[2858.00040957]
0.05	[2433.1744367]	[2852.66573517]
0.06	[2451.52849064]	[2851.33021344]
0.07	[2468.07755253]	[2852.34999406]
0.08	[2483.36564653]	[2854.87973918]
0.09	[2497.74025857]	[2858.44442115]
0.1	[2511.43228199]	[2862.75794143]
0.11	[2524.60003852]	[2867.63790917]
0.12	[2537.35489985]	[2872.96228271]
0.13	[2549.77688678]	[2878.64586939]
0.14	[2561.92452773]	[2884.62691417]
0.15	[2573.84128774]	[2890.85910969]
0.16	[2585.55987497]	[2897.30665895]
0.17	[2597.10519217]	[2903.94112629]
0.18	[2608.49640025]	[2910.73937213]
0.19	[2619.74838623]	[2917.68216413]
0.2	[2630.8728232]	[2924.75322165]
0.21	[2641.87894616]	[2931.93854417]
0.22	[2652.77412633]	[2939.22592987]

0.23	[2663.56430077]	[2946.60462378]
0.24	[2674.25429667]	[2954.06505602]
0.25	[2684.84807809]	[2961.59864341]
0.26	[2695.34893502]	[2969.19763677]
0.27	[2705.75962912]	[2976.85500119]
0.28	[2716.0825067]	[2984.56432079]
0.29	[2726.31958674]	[2992.31972181]
0.3	[2736.4726296]	[3000.11580946]
0.31	[2746.54319109]	[3007.94761559]
0.32	[2756.53266482]	[3015.81055453]
0.33	[2766.44231574]	[3023.70038563]
0.34	[2776.27330654]	[3031.61318093]
0.35	[2786.02671854]	[3039.54529713]
0.36	[2795.70356824]	[3047.49335111]
0.37	[2805.30482034]	[3055.45419817]
0.38	[2814.83139806]	[3063.42491285]
0.39	[2824.28419133]	[3071.40277169]
0.4	[2833.66406312]	[3079.38523776]
0.41	[2842.97185452]	[3087.36994673]
0.42	[2852.2083886]	[3095.35469418]
0.43	[2861.3744735]	[3103.33742413]
0.44	[2870.47090474]	[3111.31621849]
0.45	[2879.49846701]	[3119.28928746]
0.46	[2888.45793552]	[3127.25496075]
0.47	[2897.35007697]	[3135.21167941]
0.48	[2906.17565032]	[3143.15798839]
0.49	[2914.93540723]	[3151.09252966]
0.5	[2923.63009243]	[3159.01403582]
0.51	[2932.26044392]	[3166.92132421]
0.52	[2940.82719309]	[3174.81329145]
0.53	[2949.33106473]	[3182.68890838]
0.54	[2957.77277699]	[3190.54721533]
0.55	[2966.15304137]	[3198.38731777]
0.56	[2974.47256259]	[3206.20838225]
0.57	[2982.73203851]	[3214.00963255]
0.58	[2990.93215999]	[3221.79034621]
0.59	[2999.07361078]	[3229.5498512]
0.6	[3007.15706742]	[3237.28752288]
0.61	[3015.1831991]	[3245.00278108]

0.62	[3023.15266757]	[3252.69508746]
0.63	[3031.06612707]	[3260.36394297]
0.64	[3038.92422416]	[3268.00888553]
0.65	[3046.72759776]	[3275.6294878]
0.66	[3054.47687898]	[3283.22535516]
0.67	[3062.17269114]	[3290.79612376]
0.68	[3069.81564971]	[3298.34145873]
0.69	[3077.40636224]	[3305.86105245]
0.7	[3084.94542842]	[3313.354623]
0.71	[3092.43344001]	[3320.82191265]
0.72	[3099.87098085]	[3328.26268646]
0.73	[3107.25862691]	[3335.67673095]
0.74	[3114.59694628]	[3343.06385289]
0.75	[3121.88649919]	[3350.42387813]
0.76	[3129.12783807]	[3357.75665047]
0.77	[3136.3215076]	[3365.0620307]
0.78	[3143.46804472]	[3372.33989556]
0.79	[3150.56797875]	[3379.59013686]
0.8	[3157.62183137]	[3386.81266063]
0.81	[3164.63011677]	[3394.00738631]
0.82	[3171.59334168]	[3401.17424594]
0.83	[3178.51200544]	[3408.31318353]
0.84	[3185.38660008]	[3415.42415428]
0.85	[3192.21761044]	[3422.50712403]
0.86	[3199.0055142]	[3429.56206859]
0.87	[3205.75078202]	[3436.58897321]
0.88	[3212.45387757]	[3443.58783202]
0.89	[3219.11525768]	[3450.55864755]
0.9	[3225.73537241]	[3457.50143021]
0.91	[3232.31466512]	[3464.41619786]
0.92	[3238.8535726]	[3471.30297539]
0.93	[3245.35252514]	[3478.16179431]
0.94	[3251.81194665]	[3484.99269234]
0.95	[3258.23225474]	[3491.79571308]
0.96	[3264.61386081]	[3498.57090566]
0.97	[3270.95717015]	[3505.3183244]
0.98	[3277.26258207]	[3512.03802854]
0.99	[3283.53048993]	[3518.7300819]
1	[3289.7612813]	[3525.39455263]

Error graphs plotted on training and test dataset for 0.0 ≤ λ ≤ 1

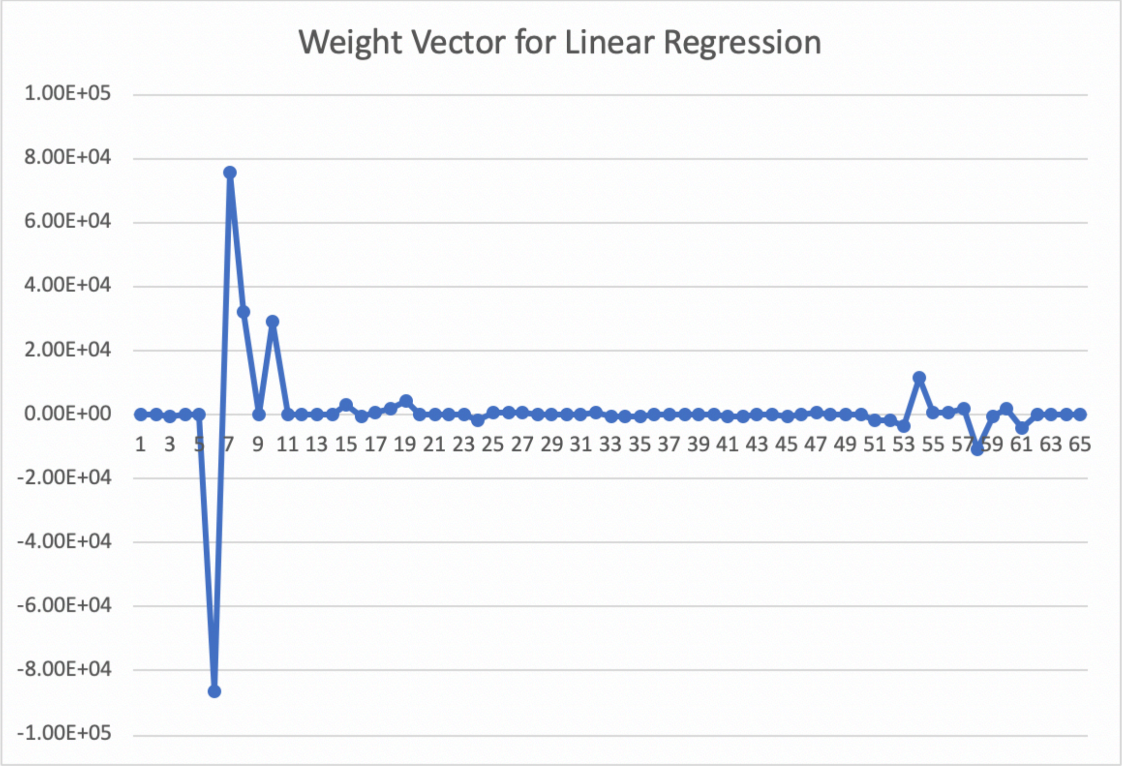


Following is observed from the above values and graphs:

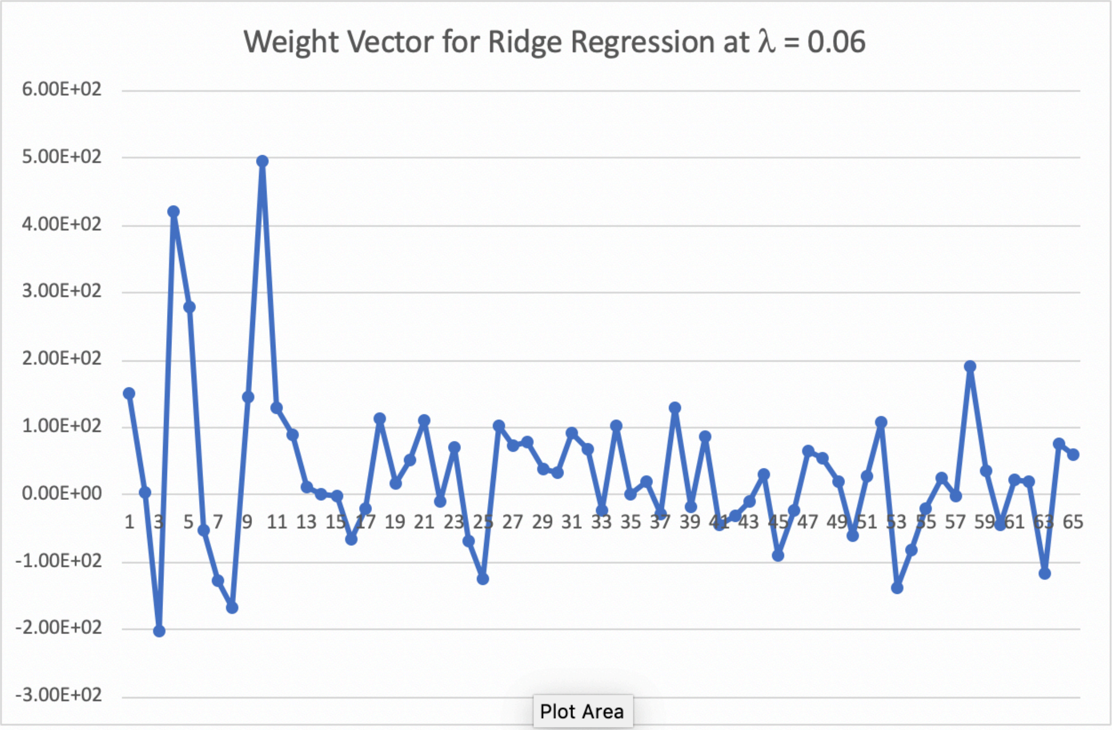
MSE for Train Data	MSE for Test Data
From the above graph, (MSE for Train Data) we can ascertain that the training data the MSE computed using learn OLE Regression is the same as MSE computed using learn Ridge Regression for **λ** is **0** i.e. no regularization is done.	From the above graph, (MSE for Test Data) we can ascertain that the error gradually increases as we increase the **λ** and the optimal value chosen for **λ** is **0.06** as it gives the least MSE.
MSE for λ = 0, when computed using learn Ridge Regression is the **same** as the MSE which is computed using learn OLE Regression.	For test data MSE for λ = 0, when computed using learn Ridge Regression is the **same** as the MSE which is computed using learn OLE Regression.

It is observed that at λ = 0.06 we get the optimal value.

Weight vectors for OLE Regression (no regularization) and Ridge Regression (regularization where λ = 0.06) are shown in the below graphs. The x-axis indicates the feature to which the weight corresponds to. The y-axis indicates the actual weight. Each data points consists of 64 features and hence we get a *65 x 1* weight vector (bias term included).

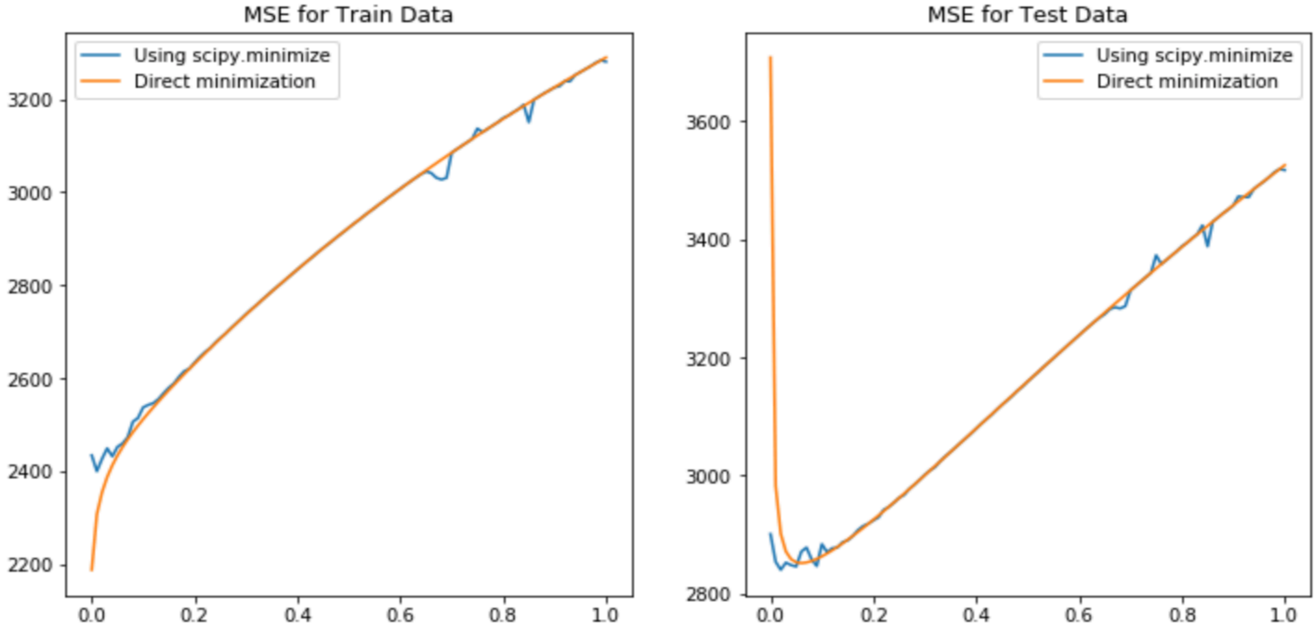


Weight Vector for Linear Regression



Weight Vector for Ridge Regression at λ = 0.06

Problem 4: Using Gradient Descent for Ridge Regression Learning


MSE for Train Data

MSE for Test Data

From the above graph, following observations can be made:

- The above two graphs which test the Mean Squared Error(MSE) for Ridge Regression computed using Gradient Descent and Ridge Regression are almost similar.
- There is also a very little deviation in the optimal λ value.
- The lines produced using Gradient Descent are **not as smooth** as the lines produced using Ridge Regression. This is due to the use of Minimize and Inverse function in Gradient Descent which tends to take more time (*inversion of matrices is expensive when the matrices are large*).

MSE for Training and Test Dataset for $0.0 \leq \lambda \leq 1.0$

Lambda	MSE Train	MSE Test
0.00	[2433.68435594]	[2900.53966835]
0.01	[2411.98791443]	[2852.47444036]
0.02	[2426.00614668]	[2839.90653503]
0.03	[2448.62539802]	[2852.13981224]
0.04	[2431.52026076]	[2847.48077078]
0.05	[2451.80707236]	[2845.54228913]
0.06	[2461.64619998]	[2874.65222402]
0.07	[2468.76506865]	[2873.63884502]
0.08	[2515.66361551]	[2862.72188406]
0.09	[2514.00286918]	[2846.40710136]
0.10	[2525.28546608]	[2862.30463042]
0.11	[2542.14824896]	[2869.809547]
0.12	[2545.9894363]	[2876.56730888]
0.13	[2555.10136434]	[2877.64817899]
0.14	[2565.52908543]	[2879.35809288]
0.15	[2579.70424584]	[2890.2458338]
0.16	[2588.58848372]	[2897.08483056]
0.17	[2602.88222]	[2907.14870323]
0.18	[2614.68452609]	[2914.06561391]
0.19	[2621.25240475]	[2918.18710993]
0.20	[2632.28109954]	[2924.25082519]
0.21	[2645.00060618]	[2928.71026975]
0.22	[2655.56137531]	[2941.44966505]
0.23	[2666.65029957]	[2951.99066418]
0.24	[2675.56033343]	[2953.56271353]
0.25	[2686.12292488]	[2961.37373688]
0.26	[2695.32216097]	[2966.85612838]
0.27	[2706.50192545]	[2977.16979675]
0.28	[2716.67653628]	[2984.14876427]

0.29	[2726.74760459]	[2992.49127926]
0.30	[2737.46439003]	[3000.85296508]
0.31	[2747.36428791]	[3008.23918343]
0.32	[2756.62242406]	[3013.83631468]
0.33	[2766.9973503]	[3023.81786968]
0.34	[2776.67826986]	[3032.22101968]
0.35	[2786.51385553	[3039.64305101]
0.36	[2795.87964668]	[3047.53985133]
0.37	[2805.02182664]	[3055.3886048]
0.38	[2814.62173325]	[3062.97380511]
0.39	[2824.16826499]	[3070.67054143]
0.40	[2833.48392372]	[3079.00120457]
0.41	[2842.98130691]	[3087.50210445]
0.42	[2852.23446303]	[3095.14124936]
0.43	[2860.4859126]	[3103.18497148]
0.44	[2870.5911012]	[3111.25495499]
0.45	[2879.73330763]	[3119.33238766]
0.46	[2888.21027505]	[3126.67218018]
0.47	[2897.02444503]	[3134.43400266]
0.48	[2906.21432713]	[3143.21738716]
0.49	[2915.12620642]	[3151.17738398]
0.50	[2923.52250197]	[3158.83528729]
0.51	[2932.24767215]	[3166.75357093]
0.52	[2940.32258493]	[3174.35760093]
0.53	[2949.3361848]	[3182.71813623]
0.54	[2957.76623668]	[3190.51525668]
0.55	[2965.80260948]	[3198.12491161]
0.56	[2974.5227136]	[3206.25208526]
0.57	[2982.74956347]	[3214.0164445]
0.58	[2990.95838909]	[3221.82812335]
0.59	[2999.13020689]	[3229.54212837]
0.60	[3007.19520684]	[3237.28993116]
0.61	[3015.33770609]	[3245.31101134]

0.62	[3023.17876246]	[3252.65310058]
0.63	[3031.0797659]	[3260.34190822]
0.64	[3038.33261643]	[3267.03008033]
0.65	[3044.32494932]	[3273.08407122]
0.66	[3041.33058101]	[3281.62196272]
0.67	[3031.02704435]	[3284.81134072]
0.68	[3027.10741055]	[3282.71760718]
0.69	[3030.65867934]	[3286.56048058]
0.70	[3084.9190029]	[3313.33396741]
0.71	[3092.46991636]	[3320.80632657]
0.72	[3099.87098731]	[3328.26423292]
0.73	[3107.28566954]	[3335.67272252]
0.74	[3114.56355236]	[3343.0133524]
0.75	[3136.868703]	[3372.83007657]
0.76	[3129.05447724]	[3357.77607252]
0.77	[3136.26885803]	[3364.99753711]
0.78	[3143.43389362]	[3372.33040777]
0.79	[3150.51640815]	[3379.52397513]
0.80	[3159.42957945]	[3387.87321288]
0.81	[3164.65929086]	[3394.01628192]
0.82	[3171.99379636]	[3401.68975636]
0.83	[3178.69876736]	[3408.45940695]
0.84	[3188.1345749]	[3423.45632676]
0.85	[3150.50369556]	[3387.95677603]
0.86	[3199.04997139]	[3429.56081853]
0.87	[3205.97433345]	[3436.6480825]
0.88	[3212.9336984]	[3443.91868627]
0.89	[3218.8772926]	[3450.05616399]
0.90	[3225.37013317]	[3457.10947351]
0.91	[3228.20744085]	[3472.5730279]
0.92	[3238.94518508]	[3471.37805479]
0.93	[3238.34883174]	[3471.20842075]
0.94	[3251.90536097]	[3485.11911195]

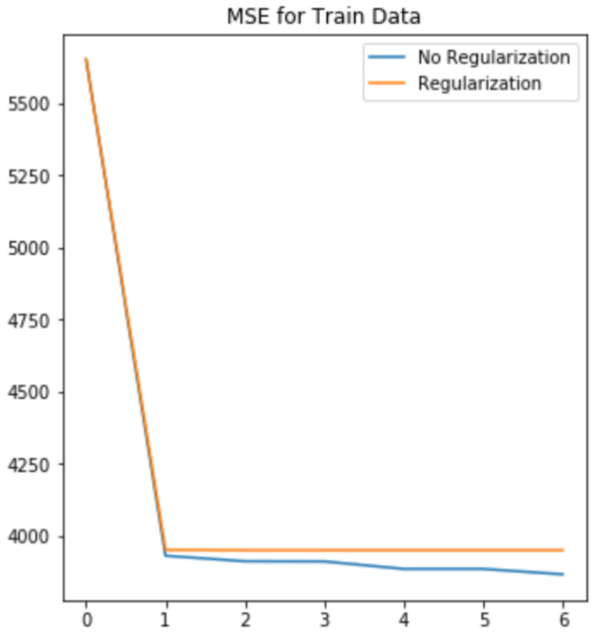
0.95	[3251.90536097]	[3491.85693074]
0.96	[3264.61512246]	[3498.08812061]
0.97	[3270.99679289]	[3505.34901888]
0.98	[3278.39972142]	[3513.51091135]
0.99	[3283.69987869]	[3518.78351357]
1.00	[3280.78343683]	[3517.19168321]

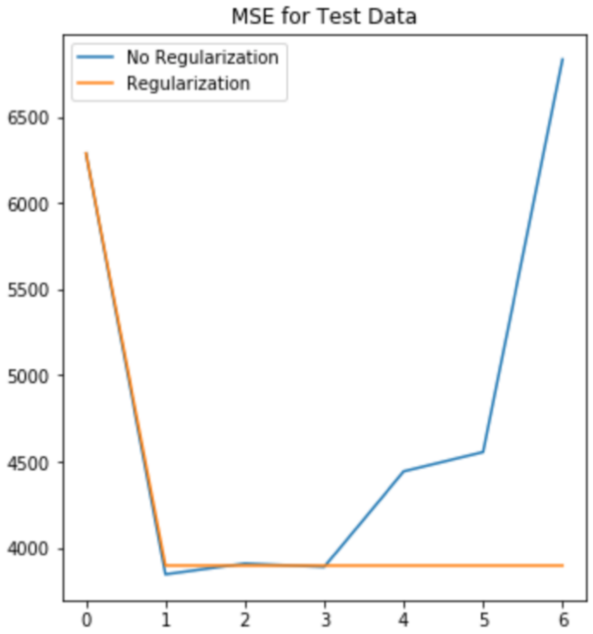
Optimal MSE using Gradient Descent for Ridge Regression:

MSE for Training data	MSE for Test data
2313.35303649	2850.64091846

Problem 5: Non-Linear Regression

The following graphs are obtained for the Training data set and the Test data set:





From the above graphs following inferences can be made:

Category	MSE FOR TRAIN DATA	MSE FOR TEST DATA
With Regularization	We can observe from the above Training data that with regularization with the increase in p the graph shows almost no difference in MSE values.	The effect of regularization is such that MSE keeps increasing as we increase p just the way it was behaving for the training data
Without Regularization	When we do not perform regularization i.e. when $\lambda = 0$, on the training data we can observe that MSE values decreases with increase in polynomial degree.	When we do not perform regularization on the Test data we can see that with increase in the polynomial degree there is a drastic increase in MSE. We can also observe that for p=1 we have least MSE and hence the **optimal value of p** is 1 without regularization.

Problem 6: Interpreting Results

Following is the result summary for the 4 methods used in this Programming Assignment:

	MSE for Train Data		MSE for Test Data	
Using Linear Regression	2187.16029493		3707.8401816	
Using Ridge Regression	λ = 0		λ = 0.6	
	2187.16029493		2851.33021344	
Using Gradient Descent	λ = 0		λ = 0.6	
	2433.6685790		2832.85210624	
Using Non-Linear Regression	With Regularization	Without Regularization	With Regularization	Without Regularization
	3895.33	6833.459	3895.856	3845.035

Conclusions Inferred:

- It can be observed that Ridge Regression is the best approach to go about all the linear models computed.
- It can be observed that the Mean Square Error is more in the Non-Linear Models as opposed to the Linear Models.
- It can be observed that the best metric to choose a setting is error. A minimum error imp,
- It can be observed that the dataset size must be accounted for when we perform a Non-Linear Regression. Right now, we are working on a small dataset but as the size of dataset increases, the running time becomes more and more important. Thus, we should perform regularization on our data when we are working with large data sets.