CSE574
Introduction to Machine Learning
Programming Assignment 1
Classification and Regression

Submitted by:
Divya J Sanghvi-50288057
Anirudh C Ramji-50290168
Saiyam Pravinchandra Shah-50291714
Team No. 78

1. Experiment with Gaussian Discriminators:

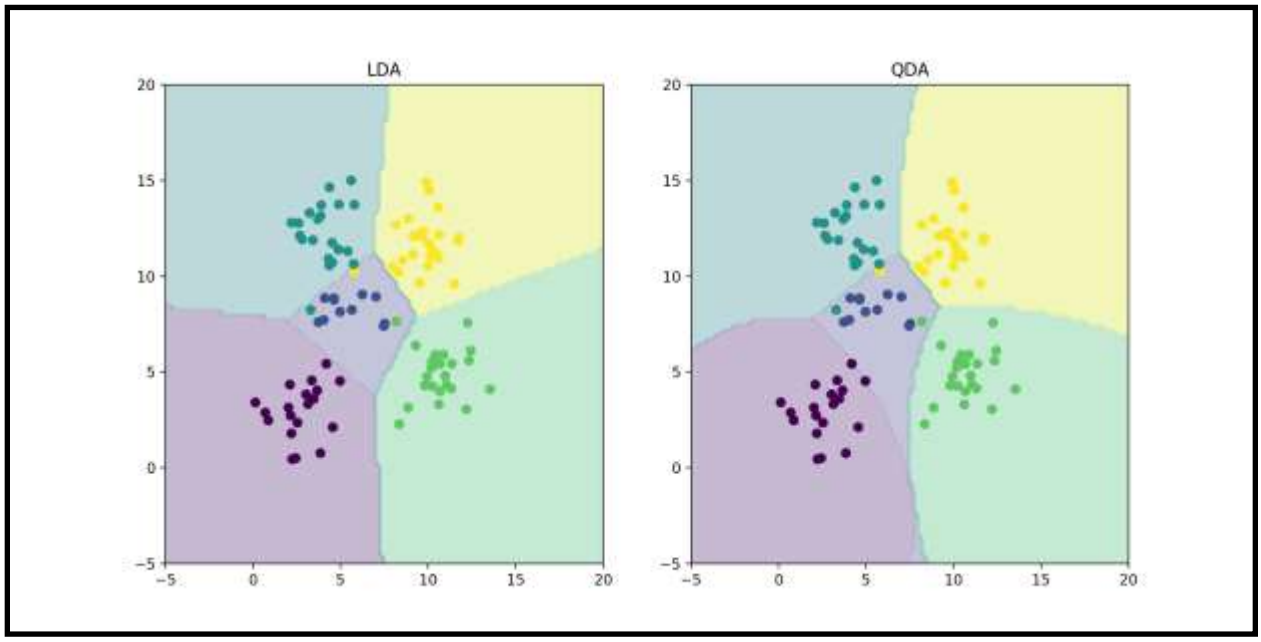

Figure 1: Output LDA and QDA

Accuracy Reported:

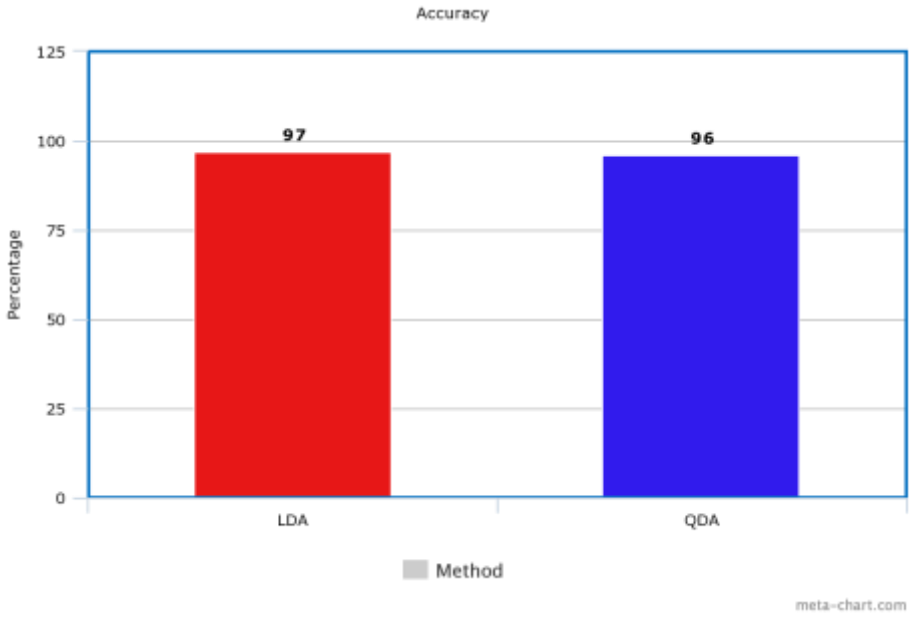

Figure 2 Accuracy Chart created with: www.meta-chart.com

For the task 1, our aim was to compare the performance of Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis which are both forms of Gaussian discriminators. We considered a uniform distribution therefore the prior probability is not a factor in estimating the probability of the test data and we only calculate the likelihood. There are two features of each input and 5 classes that the output may belong to in our dataset. In both QDA and LDA we first calculate the local mean of each of the five classes to generate the mean matrix. The difference between the two methods lies in the covariance matrix which is the key factor in plotting the boundaries of the classes. LDA assumes a common covariance matrix which means that the covariance is calculated by using the global mean of the entire data. The discriminant function produces a linear decision boundary. Whereas for QDA we calculate covariance matrix for each input data set by using the local mean of each input set. The QDA learns a quadratic decision boundary (nonlinear curves)

In general cases, LDA is much less flexible classifier than QDA - which allows for much flexible boundaries due to its nonlinear curve nature. But LDA is better than QDA if there are fewer training observations or if we have a linearly separable data. In contrast, QDA should be used when the training set is very large or when the data can't be separated linearly.

2. Experiment with Linear Regression:

	MSE without Intercept	**MSE with Intercept**
Training Data	**19099.44684457**	**2187.16029493**
Testing Data	**106775.36155731**	**3707.84018145**

Table 1: Results of Linear Regression

We observe that there is decrease of about 88 percent in error for training data and 96 percent decrease in error in testing data when intercept is considered than without intercept

We calculate the weight and the MSE by using the following formulas

$$\widehat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

It is seen that with intercept the MSE value is much smaller as compared to without intercept. The smaller the MSE, the closer we are to finding the best fit line. When we add the intercept, we are much closely aligned to the actual data. The overall accuracy is much higher with intercept than without it. This is because, without the intercept, the line would pass through origin and hence may not fit well with the data. When we add the intercept, the line that we are using to do regression moves close to the actual data. The MSE of the training data is lesser due to less outliers as compared to test data

3. Experiment with Ridge Regression:

Ridge Regression is similar to Linear Regression with an additional parameter called Regularization parameter(λ). The λ will be fed in a way to reduce the Mean squared error. Thus, this λ helps to reduce the mean squared error and hence is better. At λ = 0 Linear Regression and Ridge Regression perform similarly. But the ridge model uses the optimal λ to minimize the outliers and hence the error is lower

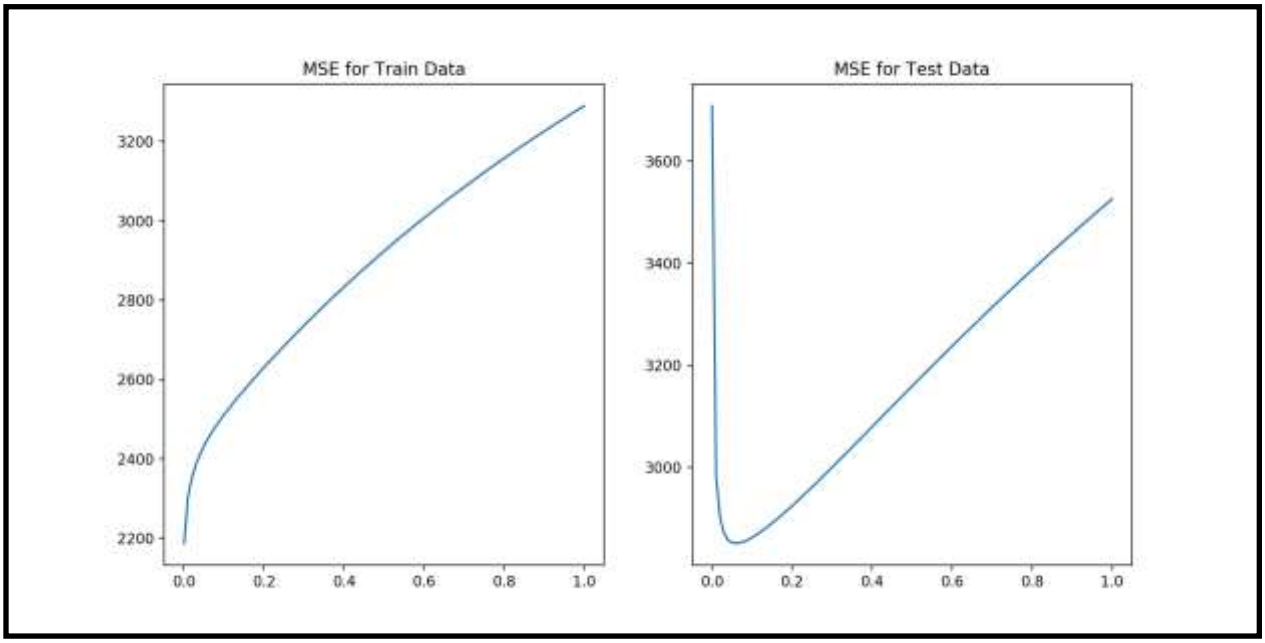

Figure 3: Result graph: Ridge Regression

λ	Test - MSE	Train -MSE	λ	Test - MSE	Train -MSE
0	3707.840181	**2187.160295**	0.3	3000.115809	2736.47263
0.01	2982.44612	2306.832218	0.31	3007.947616	2746.543191
0.02	2900.973587	2354.071344	0.32	3015.810555	2756.532665
0.03	2870.941589	2386.780163	0.33	3023.700386	2766.442316
0.04	2858.00041	2412.119043	0.34	3031.613181	2776.273307
0.05	2852.665735	2433.174437	0.35	3039.545297	2786.026719
0.06	**2851.330213**	2451.528491	0.36	3047.493351	2795.703568
0.07	2852.349994	2468.077553	0.37	3055.454198	2805.30482
0.08	2854.879739	2483.365647	0.38	3063.424913	2814.831398
0.09	2858.444421	2497.740259	0.39	3071.402772	2824.284191
0.1	2862.757941	2511.432282	0.4	3079.385238	2833.664063
0.11	2867.637909	2524.600039	0.41	3087.369947	2842.971855
0.12	2872.962283	2537.3549	0.42	3095.354694	2852.208389
0.13	2878.645869	2549.776887	0.43	3103.337424	2861.374474
0.14	2884.626914	2561.924528	0.44	3111.316218	2870.470905
0.15	2890.85911	2573.841288	0.45	3119.289287	2879.498467
0.16	2897.306659	2585.559875	0.46	3127.254961	2888.457936
0.17	2903.941126	2597.105192	0.47	3135.211679	2897.350077
0.18	2910.739372	2608.4964	0.48	3143.157988	2906.17565
0.19	2917.682164	2619.748386	0.49	3151.09253	2914.935407
0.2	2924.753222	2630.872823	0.5	3159.014036	2923.630092
0.21	2931.938544	2641.878946	0.51	3166.921324	2932.260444
0.22	2939.22593	2652.774126	0.52	3174.813291	2940.827193
0.23	2946.604624	2663.564301	0.53	3182.688908	2949.331065
0.24	2954.065056	2674.254297	0.54	3190.547215	2957.772777
0.25	2961.598643	2684.848078	0.55	3198.387318	2966.153041
0.26	2969.197637	2695.348935	0.56	3206.208382	2974.472563
0.27	2976.855001	2705.759629	0.57	3214.009633	2982.732039
0.28	2984.564321	2716.082507	0.58	3221.790346	2990.93216
0.29	2992.319722	2726.319587	0.59	3229.549851	2999.073611

0.6	3237.287523	3007.157067	0.81	3394.007386	3164.630117
0.61	3245.002781	3015.183199	0.82	3401.174246	3171.593342
0.62	3252.695087	3023.152668	0.83	3408.313184	3178.512005
0.63	3260.363943	3031.066127	0.84	3415.424154	3185.3866
0.64	3268.008886	3038.924224	0.85	3422.507124	3192.21761
0.65	3275.629488	3046.727598	0.86	3429.562069	3199.005514
0.66	3283.225355	3054.476879	0.87	3436.588973	3205.750782
0.67	3290.796124	3062.172691	0.88	3443.587832	3212.453878
0.68	3298.341459	3069.81565	0.89	3450.558648	3219.115258
0.69	3305.861052	3077.406362	0.9	3457.50143	3225.735372
0.7	3313.354623	3084.945428	0.91	3464.416198	3232.314665
0.71	3320.821913	3092.43344	0.92	3471.302975	3238.853573
0.72	3328.262686	3099.870981	0.93	3478.161794	3245.352525
0.73	3335.676731	3107.258627	0.94	3484.992692	3251.811947
0.74	3343.063853	3114.596946	0.95	3491.795713	3258.232255
0.75	3350.423878	3121.886499	0.96	3498.570906	3264.613861
0.76	3357.75665	3129.127838	0.97	3505.318324	3270.95717
0.77	3365.062031	3136.321508	0.98	3512.038029	3277.262582
0.78	3372.339896	3143.468045	0.99	3518.730082	3283.53049
0.79	3379.590137	3150.567979	1	3525.394553	3289.761281
0.8	3386.812661	3157.621831			

Table 2: MSE – Train and MSE – Test for different values of λ

The optimal value of λ is 0.06 for testing data

Why?

A) This is because at this value of λ the MSE for the test data is minimum, also at this point training and test data MSE start to converge. If we would have taken least MSE for training data as our reference for **λ,** we see through the graph that the MSE is very high at **λ = 0.0**

B) Similarly, the optimal value of **λ is 0.0** for training data

Comparing the two approaches, linear regression and Ridge regression, in terms of MSE:

Model	Data	MSE	Optimal λ
Linear Regression	Testing Data	**3707.84018103**	
Linear Regression	Training Data	**2187.16029493**	
Ridge Regression	Training Data	2187.16029493	0
Ridge Regression	Testing data	2851.330213	0.06

Table 3: Comparison of Linear and Ridge Regression

From the values of MSE for Linear regression and ridge regression it is clear that the error for ridge regression for test data is lower for ridge regression and it is a better approach. This is because of the regularization parameter.
We plotted the weights for ridge regression as well as linear regression

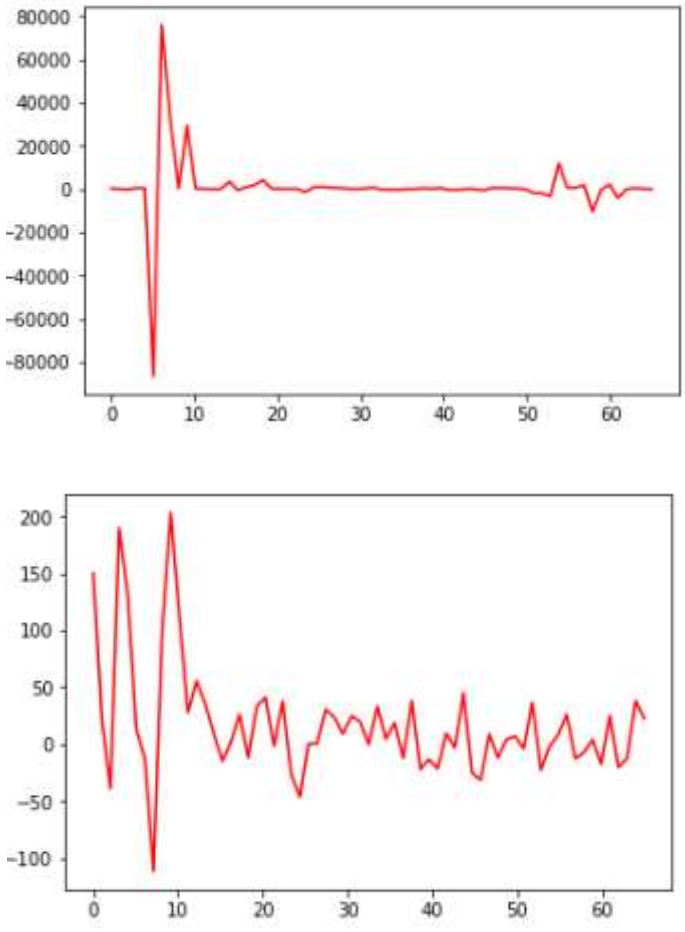

Figure 4 Weights learnt using linear regression and Ridge regression

Comparing the two approaches, in terms of **weights**, we can see that weights learnt using Linear Regression have a much higher magnitude compared to weights learnt using Ridge Regression. This huge difference is due to the regularization parameter (λ) used in Ridge Regression.

4. Experiment with Gradient Descent Ridge Regression:

In the plots above, the curves obtained using regular ridge regression and using gradient descent method is nearly identical. However, the lines produced using gradient descent method is not as smooth as those produced using regular ridge regression and has some outliers. The optimal MSE using gradient descent for ridge regression is as follows:

$$J(\mathbf{w}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) + \frac{1}{2}\lambda\mathbf{w}^\top\mathbf{w}$$

a = (y-Xw)
J(w) = 1/2.a^T.a+1/2 λ.w.w^T
J(w) = 1/2.(y-Xw)^T.(y-Xw) + 1/2. λ.w.w^T
∂ J(w)/∂ (w) = ð J(w)/∂ (a).∂ (a)/∂ (w) + λ.w
= a. ∂ (y-Xw)/∂ w + λ.w
= - X^T.(y-Xw)+ λ.w

1) As seen above and from the data got from ridge regression, the graph is very similar for gradient descent in training data and test data. However, when the number of iterations is less, the graph of gradient descent is not very smooth and has a lot of outliers. This is because the minimize function in gradient descent takes a while to converge. Hence, the regular ridge regression is faster than gradient descent method and gives lesser MSE.
2) However, when the matrices are bigger, the inversion of matrices is expensive. In such cases, gradient descent is a better option compared to regular ridge regression.

λ	MSES_Train	MSES_Test
0	[2433.66654518]	[2900.54558522]
0.01	[2462.87709387]	[2865.23321542]
0.02	**[2414.54944642]**	[2826.49430295]
0.03	[2460.63999916]	[2857.91707101]
0.04	[2445.43441372]	[2825.43233077]
0.05	[2454.68254169]	[2836.33982297]
0.06	[2452.30279696]	**[2822.88525976]**
0.07	[2465.33949178]	[2858.29592554]

0.08	[2512.24886387]	[2865.54815997]
0.09	[2516.57632066]	[2858.87542997]
0.1	[2523.62501909]	[2866.60069679]
0.11	[2540.02047857]	[2869.22199484]
0.12	[2554.2671853]	[2888.9973487]
0.13	[2553.86644424]	[2881.19430795]
0.14	[2570.8024785]	[2885.96811464]
0.15	[2574.77458541]	[2888.02214446]
0.16	[2591.9478732]	[2901.53954016]
0.17	[2601.70695583]	[2903.3733664]
0.18	[2615.32470242]	[2915.91872122]
0.19	[2625.97203266]	[2927.31814655]
0.2	[2635.2235638]	[2921.80477795]
0.21	[2644.36680497]	[2931.36219534]
0.22	[2652.81939362]	[2938.91435777]
0.23	[2665.49793367]	[2946.44359128]
0.24	[2674.31056798]	[2953.6333662]
0.25	[2685.41632263]	[2960.09457311]
0.26	[2696.1407946]	[2969.19545738]
0.27	[2706.53693348]	[2975.66633378]
0.28	[2716.76222488]	[2984.10765405]
0.29	[2726.36439953]	[2989.90616335]
0.3	[2736.6005913]	[2998.63354443]
0.31	[2746.02950586]	[3005.75720589]
0.32	[2756.86578938]	[3015.84379709]
0.33	[2767.04411358]	[3023.84583516]
0.34	[2775.74070585]	[3030.12054415]
0.35	[2786.37645617]	[3040.35690164]
0.36	[2796.12605078]	[3047.51290726]
0.37	[2805.03181613]	[3055.31125809]
0.38	[2814.57950003]	[3061.94765052]
0.39	[2824.10944483]	[3070.63328783]
0.4	[2833.95384595]	[3079.6065857]
0.41	[2842.80756028]	[3087.16686604]
0.42	[2852.43084572]	[3094.67237102]
0.43	[2861.64885976]	[3103.63170604]
0.44	[2870.5952338]	[3111.40217379]
0.45	[2878.81729934]	[3119.4338857]
0.46	[2888.53784574]	[3127.2959535]
0.47	[2897.37102685]	[3135.18398259]
0.48	[2905.65915334]	[3143.64250021]
0.49	[2914.54328488]	[3150.33843373]
0.5	[2923.60826036]	[3158.88492328]

0.51	[2932.29072835]	[3166.91044117]
0.52	[2940.89095024]	[3174.73280479]
0.53	[2949.33340361]	[3182.69121548]
0.54	[2957.4287865]	[3190.25247813]
0.55	[2966.18777656]	[3198.40980191]
0.56	[2974.47808413]	[3206.20830912]
0.57	[2982.78732701]	[3214.0389739]
0.58	[2990.9294329]	[3221.7727886]
0.59	[2999.11680572]	[3229.57385586]
0.6	[3007.17199621]	[3237.27907151]
0.61	[3015.3377061]	[3245.31101136]
0.62	[3023.17909951]	[3252.65538745]
0.63	[3031.07976651]	[3260.34191293]
0.64	[3038.33261647]	[3267.03008046]
0.65	[3044.32494906]	[3273.084071]
0.66	[3041.33058103]	[3281.62196275]
0.67	[3031.02704435]	[3284.81134072]
0.68	[3027.10741054]	[3282.71760718]
0.69	[3030.65867934]	[3286.56048058]
0.7	[3084.88885179]	[3313.27256418]
0.71	[3092.4414649]	[3320.78976894]
0.72	[3099.86819546]	[3328.24744113]
0.73	[3107.28566962]	[3335.67272282]
0.74	[3114.57009354]	[3343.00808501]
0.75	[3136.86870329]	[3372.83007697]
0.76	[3129.05447908]	[3357.77607471]
0.77	[3136.29809764]	[3365.06829146]
0.78	[3143.55518029]	[3372.38367726]
0.79	[3150.57282836]	[3379.6516741]
0.8	[3159.42958063]	[3387.87321409]
0.81	[3164.56604315]	[3393.97509754]
0.82	[3171.99379636]	[3401.68975625]
0.83	[3178.69371626]	[3408.4447849]
0.84	[3188.13456913]	[3423.45632173]
0.85	[3150.50369555]	[3387.95677602]
0.86	[3199.05648301]	[3429.57543966]
0.87	[3205.97433642]	[3436.64808618]
0.88	[3212.93369837]	[3443.91868623]
0.89	[3218.87729258]	[3450.05616388]
0.9	[3225.3780215]	[3457.17571633]
0.91	[3228.20744089]	[3472.5730281]
0.92	[3238.85403659]	[3471.2765989]
0.93	[3238.34883184]	[3471.20842085]

0.94	[3251.90470284]	[3485.11610792]
0.95	[3258.26773583]	[3491.85672487]
0.96	[3264.61510931]	[3498.08810704]
0.97	[3271.00645915]	[3505.36727756]
0.98	[3278.39972408]	[3513.5109151]
0.99	[3283.69987871]	[3518.78351379]
1	[3280.78343654]	[3517.19168302]

Table 4: Training and testing errors using gradient descent

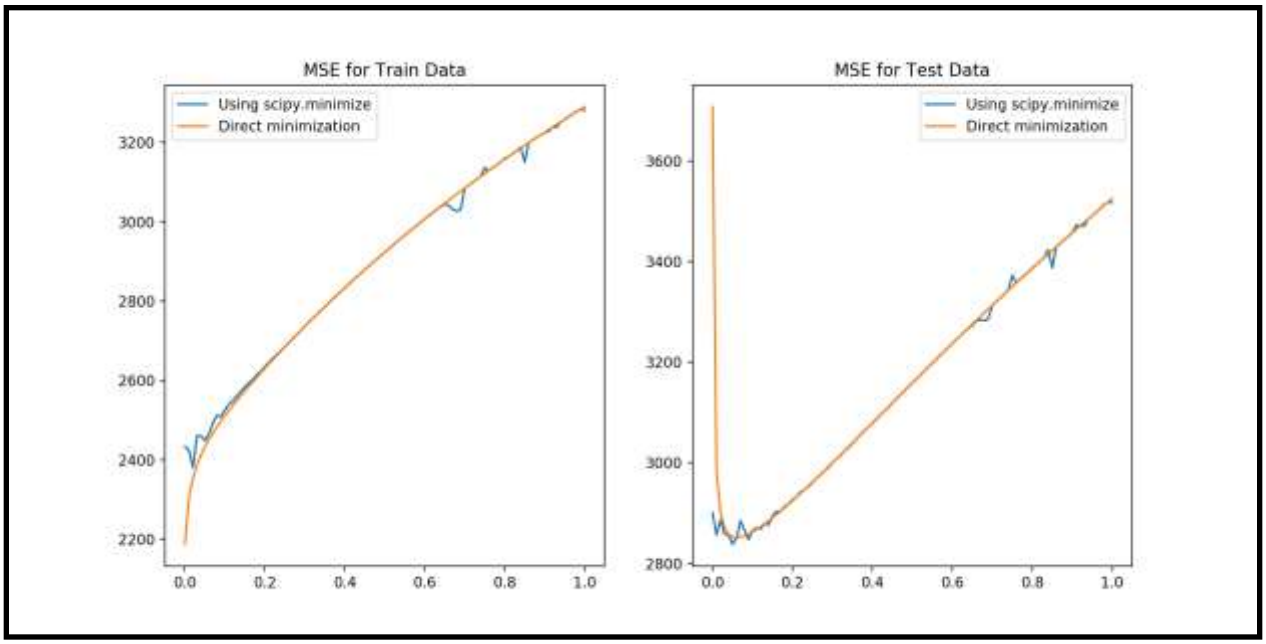

Figure 5: Final Result graph - iterations 20

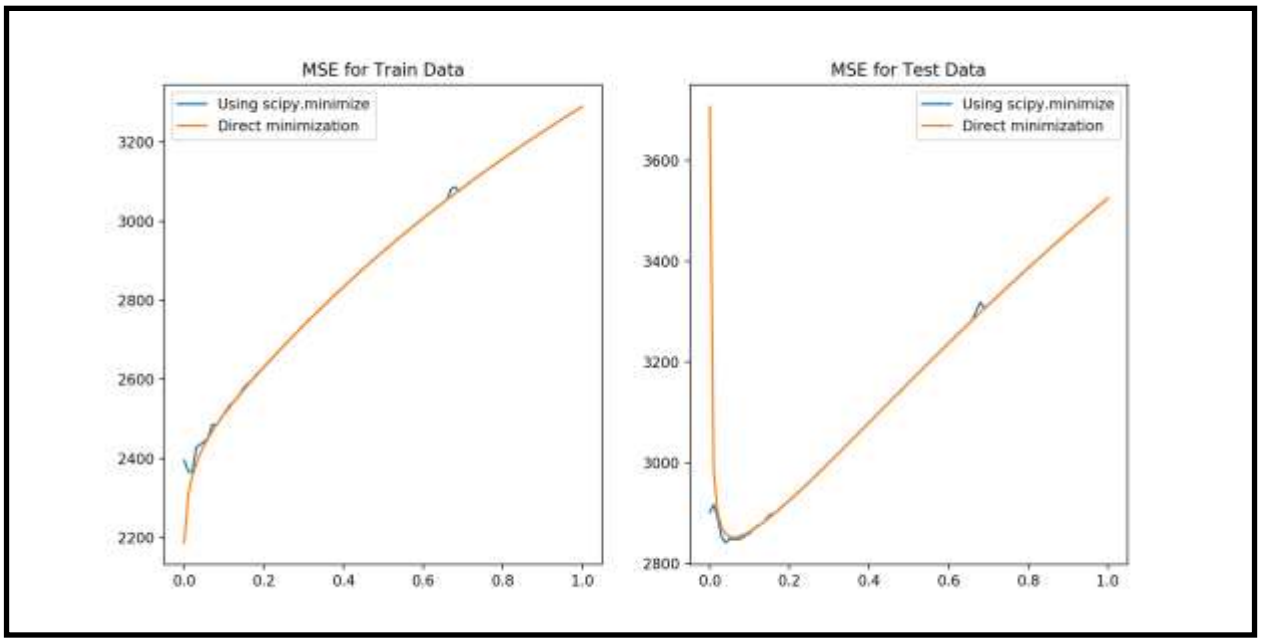

Figure 6 Final Result Graph - No of iterations 30

As the number of iterations increase the graph becomes more and more smoother for gradient descent in Ridge Regression.

5. Non-Linear Regression

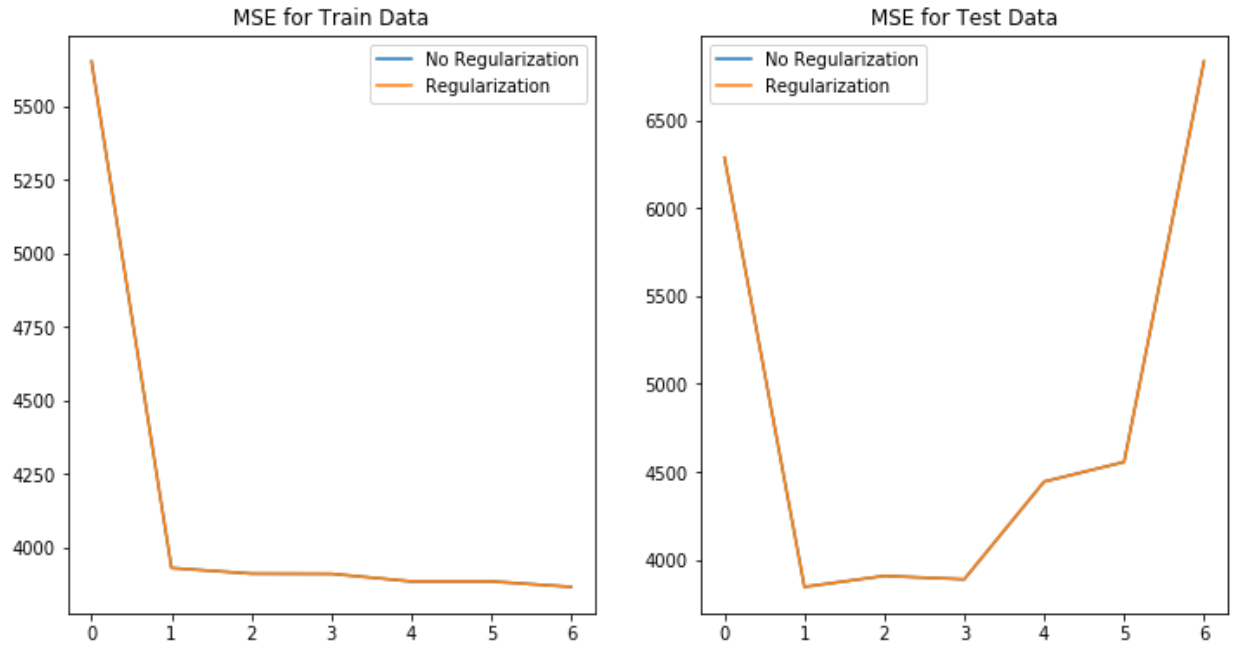

Figure 7 : λ = 0 Linear Regression Graph

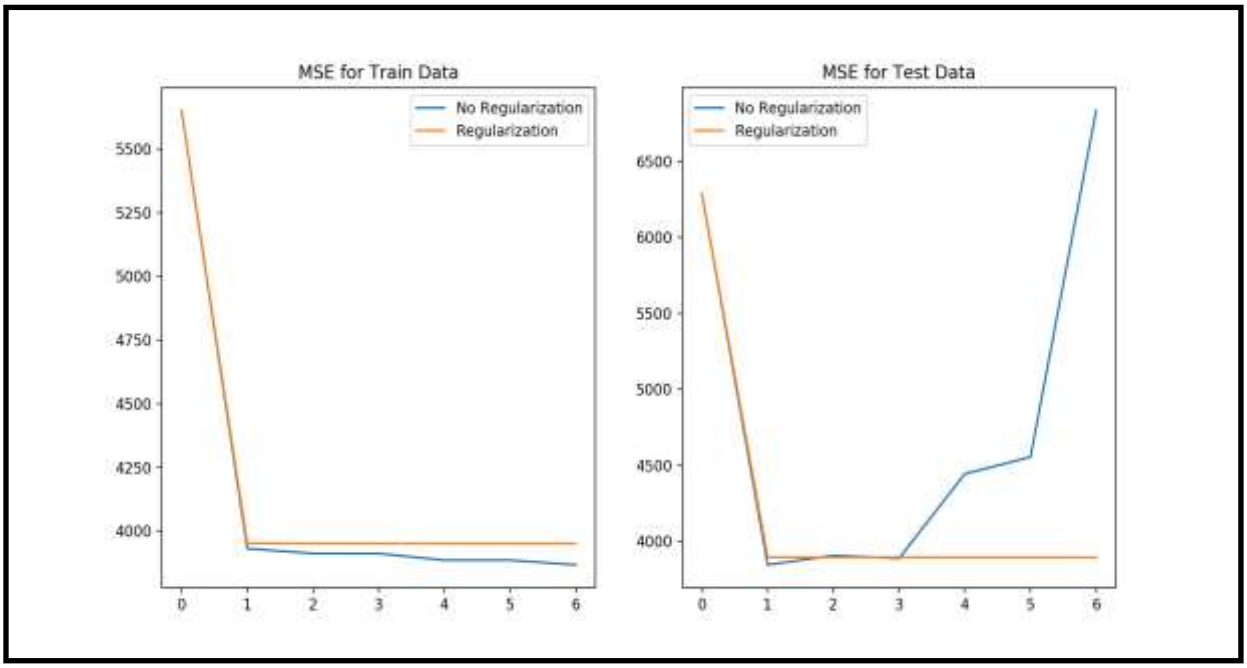

Figure 8: λ = 0.06 Non-Linear regression Result Graph

We vary the degree of the polynomial to check for various results. Also, the analysis is performed using **λ = 0** as well as **λ = 0.06**

Results for **λ = 0.06 for training data**

p	Error without regularization	Error with regularization
0	5650.71053889762	5650.711907032113
1	3930.9154073158998	3951.8391235601043
2	3911.839671204953	3950.68731237552
3	**3911.18866493145**	3950.682531518713
4	3885.4730681122714	3950.68233679537
5	3885.40715739708	3950.6823351770204
6	3866.883449446049	**3950.682335142783**

Table 4: Comparison of MSE with and without regularization using different values of p

Results of error for **λ** = 0 for training data

p	Error
0	5650.71053889762
1	3930.9154073158998
2	3911.839671204953
3	3911.18866493145
4	3885.4730681122714
5	3885.40715739708
6	**3866.883449446049**

Table 5: ***Comparison of MSE with and without regularization using different values of p***

1) We can clearly observe that the error is constantly decreasing with the value of increasing p and is optimal at p = 6
2) Also, for λ = 0 there is no regularization term. It also follows the same pattern that the error decreases when the value of p increases

Results of error for **λ** = 0.06 for testing data

p	Error without regularization	Error with regularization
0	6286.404791680893	6286.88196694145
1	**3845.034730173412**	3895.8564644739618
2	3907.1280991079357	3895.584055938918
3	3887.975538236018	**3895.582715923101**
4	4443.327891813364	3895.582668283524
5	4554.83037743454	3895.58266870442
6	6833.4591487189755	3895.5826687190947

Table 6: ***Comparison of MSE with and without regularization using different values of p using testing data***

Results of error for **λ** = 0. for testing data

p	Error
0	6286.404791680893
1	**3845.034730173412**
2	3907.1280991079357
3	3887.975538236018
4	4443.327891813364
5	4554.83037743454
6	6833.4591487189755

Table 7: ***Comparison of MSE with and without regularization using different values of p using testing data***

1) We can observe that the error decreases till the value p =1 then increases and then decreases again at p=3 and then it again starts increasing till p =6 without regularization whereas it remains constant with regularization which plays the role of decreasing the magnitude
2) Also, for λ = 0 there is no regularization term. It also follows the same pattern.
3) When we compare the MSE with and without regularization term, the MSE with regularization term is higher which means the function does not perform overfitting and the regularization term in fact increases the error

Thus, the optimal value for p = 6 in training set whereas it is p =1 for testing data set

6. Interpreting Results

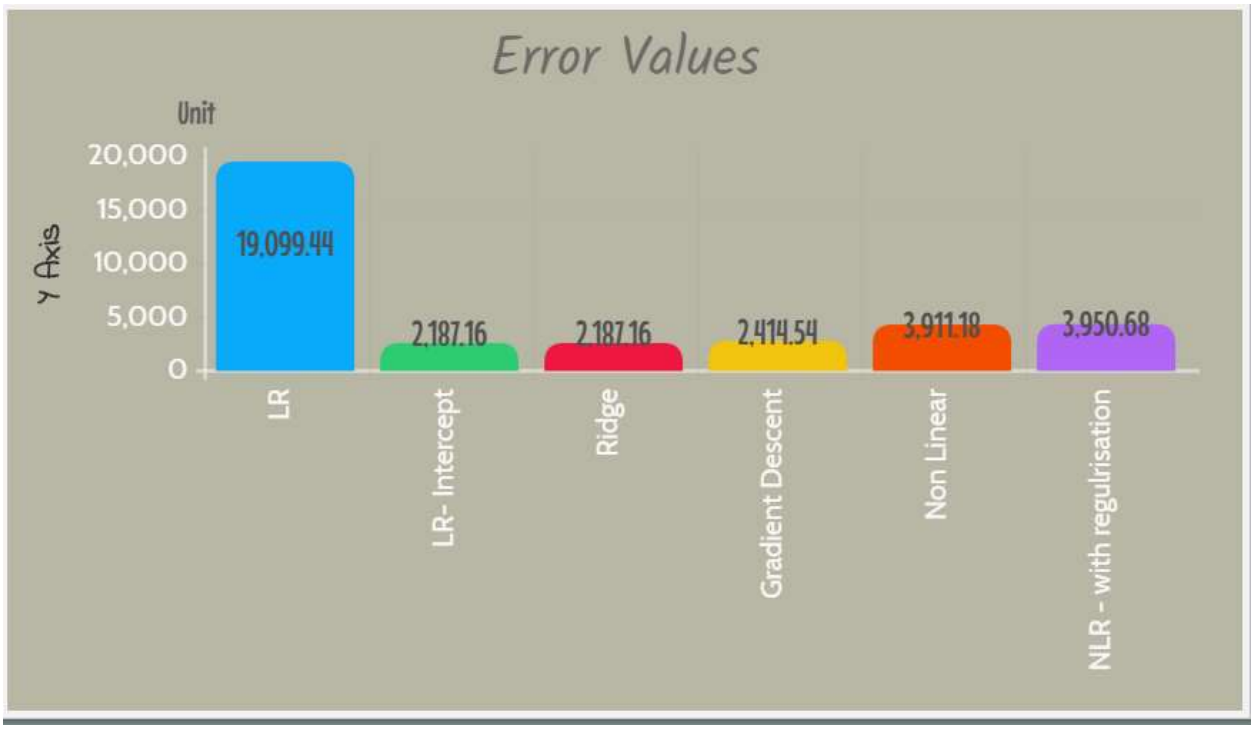

Figure 9: Training Data

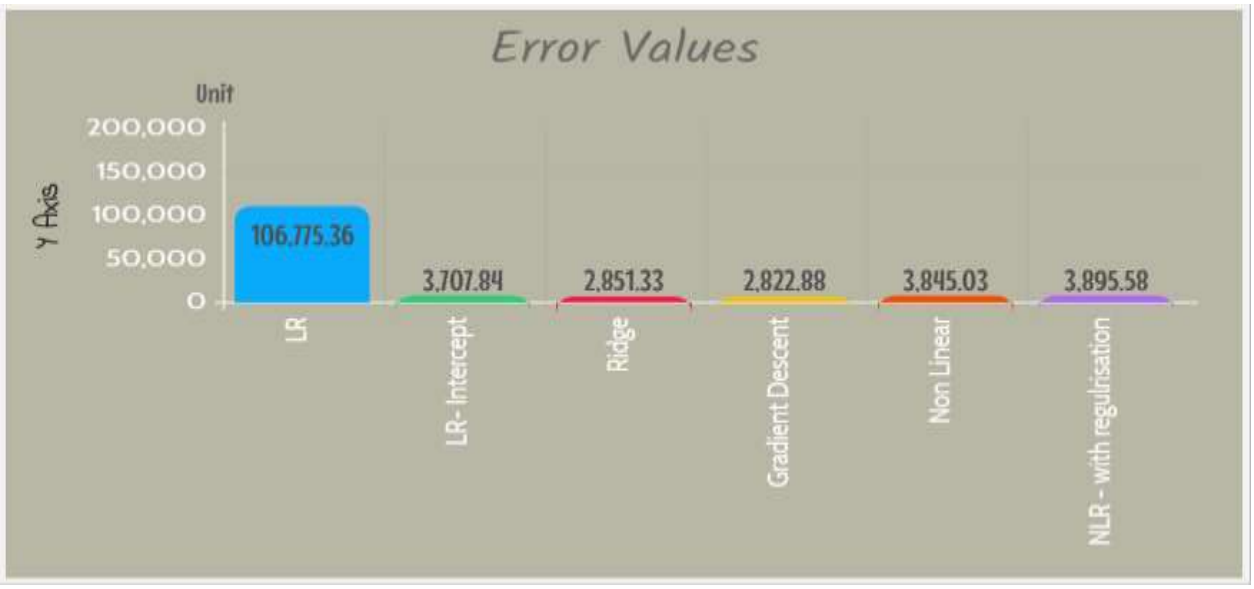

Figure 10: Testing Data

The best metric that we can use to choose the best setting from the different approaches are the accuracy value obtained for each of the approaches. We can calculate the accuracy of each approach by comparing the MSE value obtained from the execution of the code. According to the table given above, the MSE value for Linear Regression with intercept is extremely high and will lead to a lot of errors, so it is an infeasible approach if we consider accuracy as our metric. We see that the Ridge Regression approach provides us with much lower values for MSE which denotes that the error obtained from this approach is the lower. Also, the MSE values for Ridge Regression with Gradient Descent are almost identical to the Ridge Regression values. However, if we perform regularization using an optimal value for λ, we see that the MSE value is at its lowest, which means that in this case we achieve optimal accuracy results. Similarly, for the Non-linear Regression approach, both for with regularization and without regularization, we see that the MSE values are slightly higher than those of the Ridge Regression approach, so we will not choose this approach either. Therefore, if we were to consider accuracy as the only metric, we will choose the Ridge Regression with Gradient Descent approach, with regularization using optimal value for λ, as it produces the highest accuracy.