CSE 574: Introduction to Machine Learning

Project Report: Classification and Regression

Kshitij Kumar | 50610480 | kkumar8@buffalo.edu

Amritesh Kuraria | 50598180 | akuraria@buffalo.edu

Instructor's Name: Dr. Asif Imran

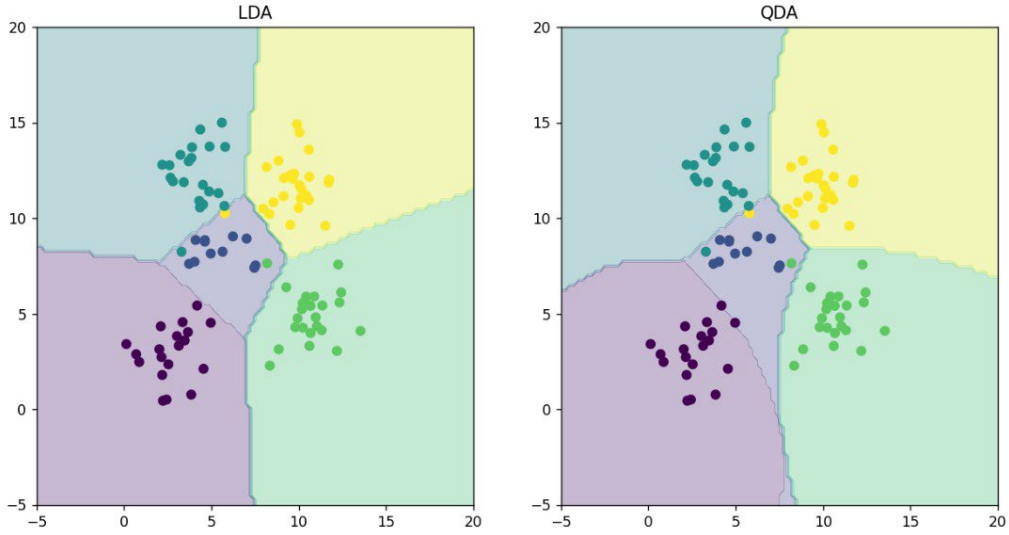
Problem 1: Experiment with Gaussian Discriminators

- **Linear Discriminant Analysis (LDA):**
  - In the ldaLearn method, I have computed the mean vectors for each class, and I have computed a covariance matrix that is common across all the classes.
  - In the ldaTest method, these parameters to classify the test data and calculate the accuracy.
  - The accuracy that we achieved using LDA is: 0.97.
- **Quadratic Discriminant Analysis (QDA):**
  - In the qdaLearn method, I have computed mean vectors and covariance matrix that are specific to each class.
  - In the qdaTest method, I am using these parameters for classification and calculating accuracy.
  - The accuracy that we achieved using QDA is: 0.96.

Plotting:

LDA Boundary: Upon plotting, we can see that the decision boundaries are linear. This assumes equal covariance across all classes, resulting in linear separation. Points in the bottom left are clustered more widely, and since the covariance in LDA is same for all the classes, the accuracy is higher.

QDA Boundary: Upon plotting, we can see that the decision boundaries are quadratic. QDA uses different covariance structures for each class, so the separations are non-linear. We can see here that the data points in bottom left are misclassified, are the accuracy is reduced for a bit.



Comparison and Explanation:

- The boundaries between LDA and QDA differ because LDA has an equal covariance across all the classes, so the boundaries are more linear. So, the boundaries may underfit the data if there are more complex distributions. On the other hand, in QDA the covariance matrix differ class to class. So, here the boundaries are more curved, and QDA can handle those data better that have complex distribution amongst themselves.
- Here, we can see that LDA performs better than QDA, suggesting that the dataset is more likely to be linearly separable. Even though QDA can detect quadratic boundaries, this improvement is unnecessary, and it can be noticed by the decreased accuracy.

Problem 2: Experiment with Linear Regression

Here, we are comparing two OLS Regression models: one that has an intercept, and the other that does not. The model that does not have any intercept/bias assumes that the model is passing through the origin, whereas the model that uses intercept is used to make sure that it accounts for any offset in the data.

We have implemented the Ordinary Least Squares (OLS) regression using the formula:

$$w = (X^T X)^{-1} X^T y$$

For the model that has an intercept, we are adding a column for including the bias and are then calculating the MSE for both training and testing dataset. Below table shows this:

	Training MSE	**Test MSE**
Without Intercept	19099.4468445708	106775.3615151266
With Intercept	2187.160294930391	3707.8401819303413

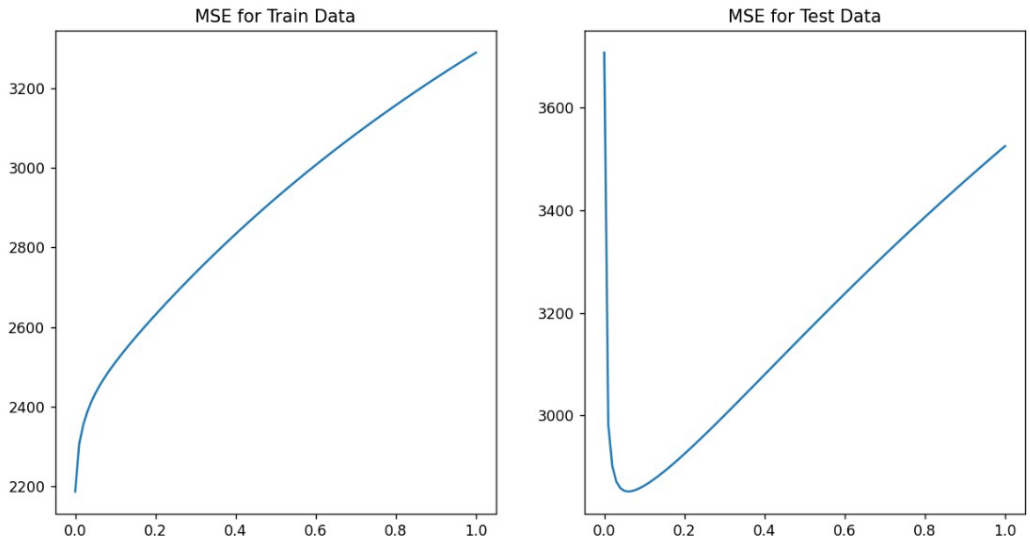
In the above result, we can see that the model with an intercept has performed much better than the one that did not have one as it has achieved a lower MSE in both Training and testing dataset.

Problem 3: Experiment with Ridge Regression

In this problem, we are using Ridge regression to compute weights for linear regression while regularizing the model to avoid overfitting. The regularization strength was controlled by the parameter λ, which was varied between 0 (no regularization) and 1, in increments of 0.01. The training and test MSEs were computed for each value of λ using the **testOLERegression** function.

Results and observations:

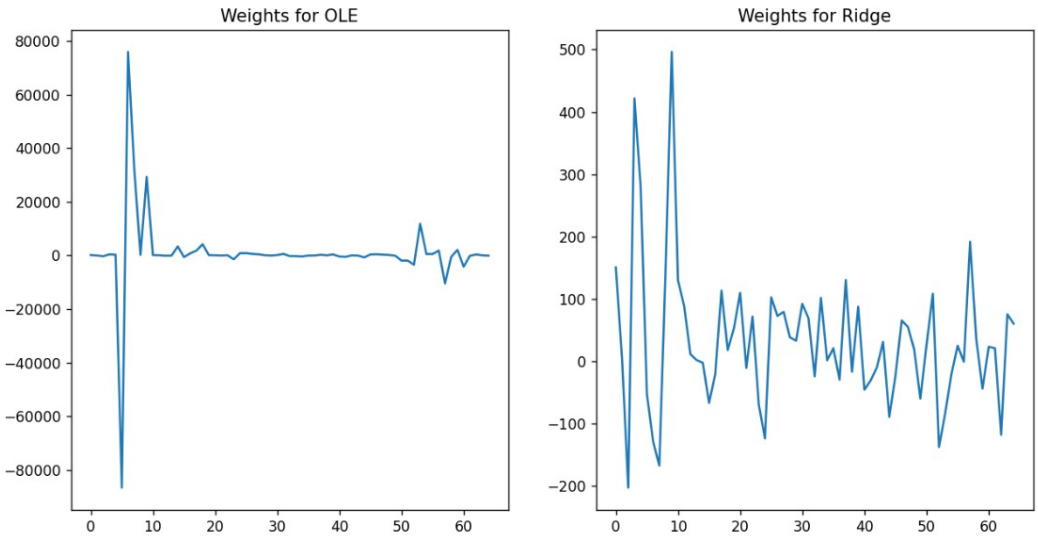
Upon varying λ from 0 to 1, we found out that the optimal value of λ is 0.06, as the Test MSE is the lowest at 2851.3302 at λ = 0.06. The results are shown in both graphical and tabular format, and the optimal λ is highlighted.



Comparison between OLE and Ridge:

We can see that upon using OLE, the Test MSE is 3707.8401819303413 while using Ridge Regression, the best Test MSE is 2851.3302 when λ = 0.06. In Ridge Regression, we are introducing a regularization term, which controls the weights, hence leading to better generalization as compared to OLE. λ = 0 would make it similar to OLE and λ = 1 would lead to over-regularization and it might reduce model's capacity to capture important details.

Comparison for weights:



In the above plot, we can see that for OLE, the values of weights are huge and fluctuate a lot (-80000 to 80000), but in the case of ridge regression, weights are comparatively small, and the fluctuations are also not very huge (-200 to 500). From this, we can conclude that in weights are penalised very heavily in ridge regression.

Optimal λ

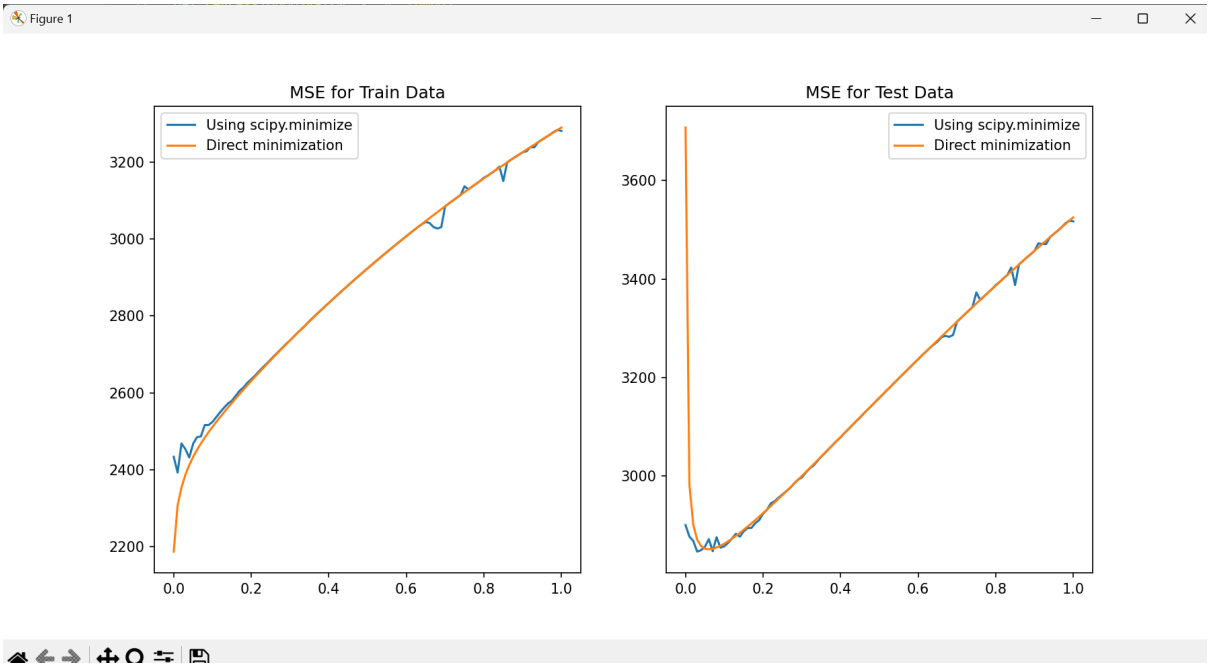
The optimal λ is 0.06 as we get the least Test MSE at this value.

Lambda	Train MSE	Test MSE	Lambda	Train MSE	Test MSE
0.00	2187.1603	3707.8402	0.25	2684.8481	2961.5986
0.01	2306.8322	2982.4461	0.26	2695.3489	2969.1976
0.02	2354.0713	2900.9736	0.27	2705.7596	2976.8550
0.03	2386.7802	2870.9416	0.28	2716.0825	2984.5643
0.04	2412.1190	2858.0004	0.29	2726.3196	2992.3197
0.05	2433.1744	2852.6657	0.30	2736.4726	3000.1158
0.06	2451.5285	2851.3302	0.31	2746.5432	3007.9476
0.07	2468.0776	2852.3500	0.32	2756.5327	3015.8106
0.08	2483.3656	2854.8797	0.33	2766.4423	3023.7004
0.09	2497.7403	2858.4444	0.34	2776.2733	3031.6132
0.10	2511.4323	2862.7579	0.35	2786.0267	3039.5453
0.11	2524.6000	2867.6379	0.36	2795.7036	3047.4934
0.12	2537.3549	2872.9623	0.37	2805.3048	3055.4542
0.13	2549.7769	2878.6459	0.38	2814.8314	3063.4249
0.14	2561.9245	2884.6269	0.39	2824.2842	3071.4028
0.15	2573.8413	2890.8591	0.40	2833.6641	3079.3852
0.16	2585.5599	2897.3067	0.41	2842.9719	3087.3699
0.17	2597.1052	2903.9411	0.42	2852.2084	3095.3547
0.18	2608.4964	2910.7394	0.43	2861.3745	3103.3374
0.19	2619.7484	2917.6822	0.44	2870.4709	3111.3162
0.20	2630.8728	2924.7532	0.45	2879.4985	3119.2893
0.21	2641.8789	2931.9385	0.46	2888.4579	3127.2550
0.22	2652.7741	2939.2259	0.47	2897.3501	3135.2117
0.23	2663.5643	2946.6046	0.48	2906.1757	3143.1580
0.24	2674.2543	2954.0651	0.49	2914.9354	3151.0925
			0.50	2923.6301	3159.0140

Lambda	Train MSE	Test MSE
0.51	2932.2604	3166.9213
0.52	2940.8272	3174.8133
0.53	2949.3311	3182.6889
0.54	2957.7728	3190.5472
0.55	2966.1530	3198.3873
0.56	2974.4726	3206.2084
0.57	2982.7320	3214.0096
0.58	2990.9322	3221.7903
0.59	2999.0736	3229.5499
0.60	3007.1571	3237.2875
0.61	3015.1832	3245.0028
0.62	3023.1527	3252.6951
0.63	3031.0661	3260.3639
0.64	3038.9242	3268.0089
0.65	3046.7276	3275.6295
0.66	3054.4769	3283.2254
0.67	3062.1727	3290.7961
0.68	3069.8156	3298.3415
0.69	3077.4064	3305.8611
0.70	3084.9454	3313.3546
0.71	3092.4334	3320.8219
0.72	3099.8710	3328.2627
0.73	3107.2586	3335.6767
0.74	3114.5969	3343.0639
0.75	3121.8865	3350.4239
0.76	3129.1278	3357.7567
0.77	3136.3215	3365.0620
0.78	3143.4680	3372.3399
0.79	3150.5680	3379.5901
0.80	3157.6218	3386.8127
0.81	3164.6301	3394.0074
0.82	3171.5933	3401.1742
0.83	3178.5120	3408.3132
0.84	3185.3866	3415.4242
0.85	3192.2176	3422.5071
0.86	3199.0055	3429.5621
0.87	3205.7508	3436.5890
0.88	3212.4539	3443.5878
0.89	3219.1153	3450.5586
0.90	3225.7354	3457.5014
0.91	3232.3147	3464.4162
0.92	3238.8536	3471.3030
0.93	3245.3525	3478.1618
0.94	3251.8119	3484.9927
0.95	3258.2323	3491.7957
0.96	3264.6139	3498.5709
0.97	3270.9572	3505.3183
0.98	3277.2626	3512.0380
0.99	3283.5305	3518.7301
1.00	3289.7613	3525.3946

Problem 4: Using Gradient Descent for Ridge Regression Learning

In this task, we are using gradient descent to estimate the ridge regression weights. Below is the plot of MSE for both Training and Testing data using scipy.minimize and Direct Minimization.



We can see that the results of Problem 4 are very similar to the results of Problem 3. Upon using the gradient descent, we found that the optimal value for λ is 0.03 which is slightly lower than that λ = 0.06 that we got in Problem 3.

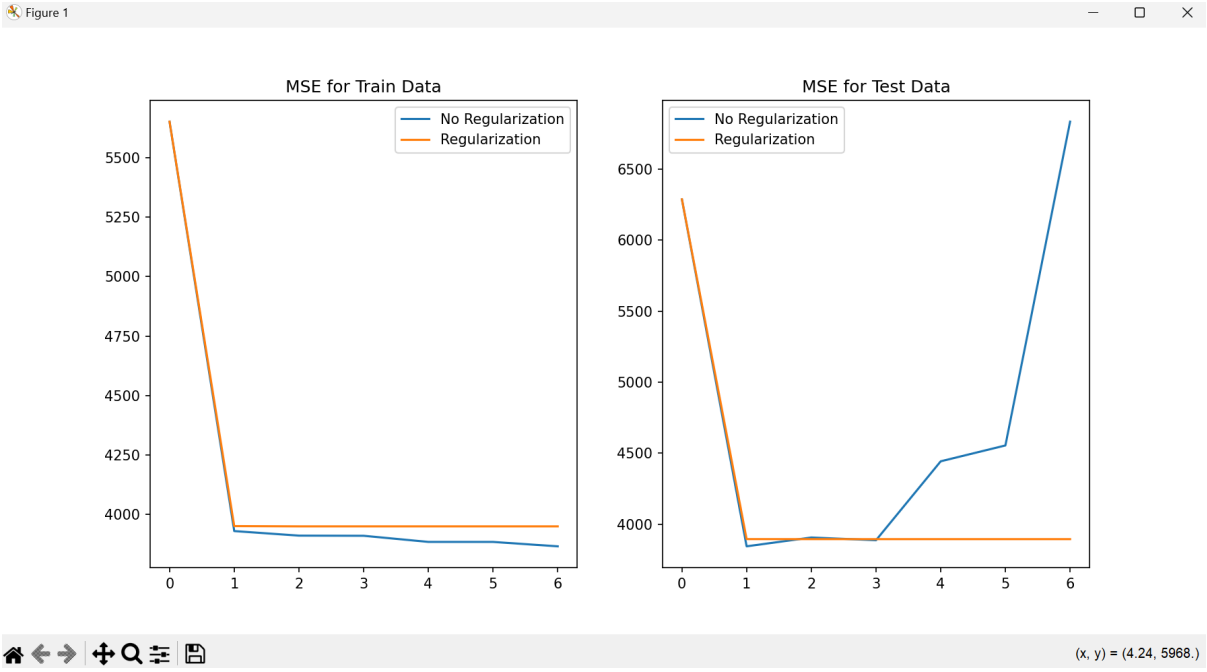
Comparison:

- At this value of λ, the Minimum Test MSE came out to be 2846.63, which is slightly lower than 2851.3302 that we got in Problem 3. This marginal improvement in Test MSE may be because of finer adjustments in the weights. Both these methods follow a very similar trend – in both the methods, the test errors start to decrease and then keep on increasing because of over-regularization.
- The problem 3's solution is computing the weights by inverting the matrix. This approach is fine as long as we have small dataset, but it can quickly get computationally expensive as the dataset grows larger. The gradient descent method that we have used in this problem scales better with larger datasets, although it requires taking care of tuning the hyperparameters.

Problem 5: Non-linear Regression

In this task, we investigate the impact of non-linear transformations on ridge regression. So, we analyse if increasing the complexity also increases the performance of the model. Here, we are calculating the error for two cases – without regularization and with regularization by using the optimal value of λ that we found in problem 3, i.e. λ = 0.06.

In the below plot, we can see the MSE for train and test data for both the cases - one that uses Regularization and the other that doesn't.



Results:

1. **Training MSE**

For λ = 0, the training MSE decreases consistently as p increases from 0 to 6, and we get the lowest training MSE when p=6. This is happening, because the model is more complex at p=6 and it is finding every detailed pattern, and thus, it fits the training data much better.

For the regularized model that has λ = 0.06, the training MSE remains a constant for p>=1. Regularizations stops the model from overfitting the dataset for higher order terms.

2. **Testing MSE**

In the above plot, we can see that for λ=0, the test MSE decreases until p=3 but then increases very sharply for p>3. This sharp increase shows that the model is overfitting, as the model is also including noise in the training data, and it is not able to do proper generalization for the test data.

In the plot, we can see that for λ=0.06, the test MSE remains nearly constant from p=1to p=6. This is because regularization doesn't let the addition of higher order polynomials change the model's performance much and it and it leads to much better generalization and consistent test performance than the unregularized models.

So, when λ=0, then p = 3 gives the best performance, and when λ=0.06, then p=1 gives the best performance, suggesting that there is no need to over-complicate the model when adding regularization.

Problem 6: Interpreting Results

The below plot shows the MSE for different higher order polynomials

Problem	Model	Training Error with metric	Testing Error with metric	Comments	Best Model/Hyper parameter
1	LDA	NA	Accuracy - 97%	LDA performs better because it linearly separates the data.	LDA
	QDA	NA	Accuracy – 96%	QDA makes complex quadratic boundaries, so it makes the model more flexible.	
2	OLS Regression (No Intercept)	MSE - 19099.4468445708	MSE - 106775.3615151266		OLS Regression with Intercept
	OLS Regression (Intercept added)	MSE - 2187.160294930391	MSE - 3707.8401819303413	Test MSE is greatly reduced by adding Intercept	
3	Ridge Regression	For λ = 0.06, MSE - 2451.5285	For λ = 0.06, MSE - 2851.3302	The testing MSE is the least when λ = 0.06	λ = 0.06
4	Direct Minimization	It varies as λ varies	The results are very similar to that of gradient descent. The error is slightly reduced.	Here, we find the optimal solution by inverting the matrix. So, it works well if the matrix is small	λ = 0.06
	Gradient Descent	It varies as λ varies	Here we get similar results as Direct Minimization. There is a very slight difference.	Here, we get similar performance to that of direct minimization, but this approach is more useful for bigger dataset	λ = 0.03
5	Non -linear regression (No	It varies as p varies. The train	Here, the test MSE varies, and the Test MSE first	Since there is no regularization, so the test error	

	Regularization)	MSE gets lower as p increases.	decreases, but then increases after p = 3	increases a lot after a reaching a threshold.	
	Non – linear regression (Regularization)	It varies as p varies, The Train MSE lowers when p = 1 and then remains constant	Here, the test MSE decreases till p = 1and then remains a constant.	Since we are using regularization here, so the test error doesn't change much. So, the model can have higher complexity and doesn't overfit.	

Recommendation:

Based on the findings, we would recommend using **Ridge Regression with gradient descent** and a **regularization parameter equal to 0.03**. And based on the above findings, we would recommend MSE for the metric.