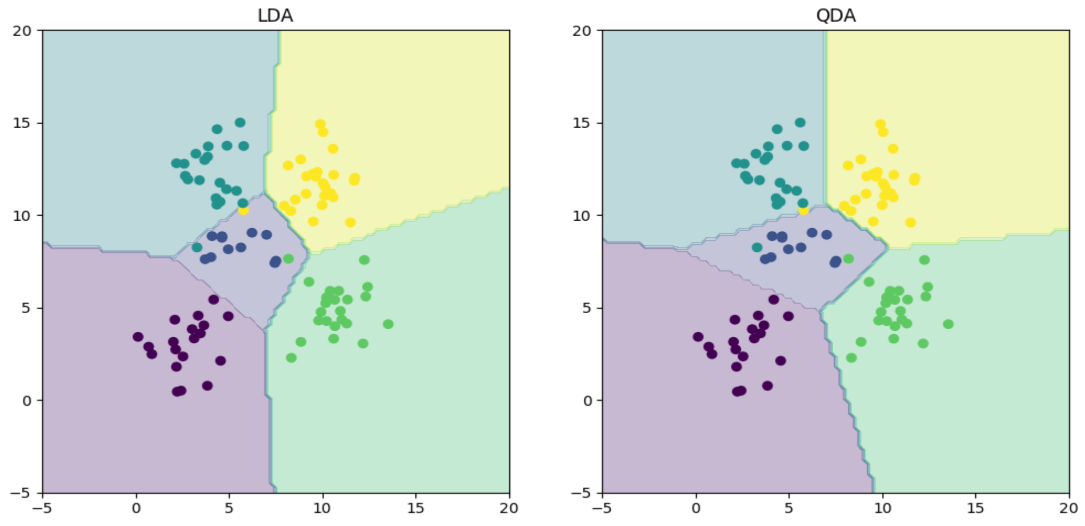
CSE 574: Introduction to Machine Learning
Programming Assignment 2

Classification and Regression

Submitted by
Group #30
Amey Mhaskar
Gideon Paul
Julien Kann

Problem 1: Experiment with Gaussian Discriminators

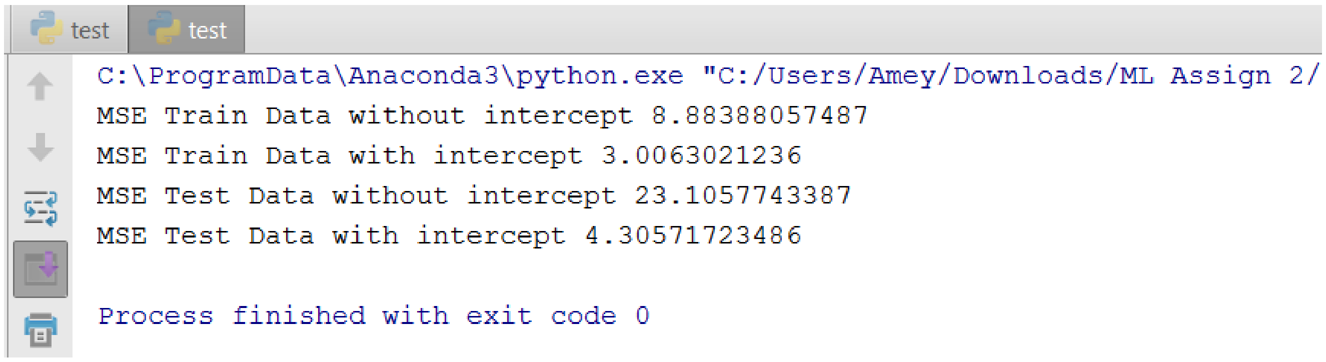
There are differences in the discriminating boundaries between Linear Discriminant Analysis(LDA) and Quadratic Discriminant Analysis(QDA).



In LDA, one covariance matrix is created and applied to all classes. The application of one common covariance matrix to every class creates linear boundaries. In QDA, a covariance matrix is generated for each class and only applied to that class. Because each class has its own separate covariance matrix applied to it creates quadratic decision boundaries.

Problem 2: Experiment with Linear Regression

The Mean Square Error(MSE) for training and testing data with and without intercepts have been calculated using linear regression. The respective MSE values are as shown in the below figure



```
C:\ProgramData\Anaconda3\python.exe "C:/Users/Amey/Downloads/ML Assign 2/
MSE Train Data without intercept 8.88388057487
MSE Train Data with intercept 3.0063021236
MSE Test Data without intercept 23.1057743387
MSE Test Data with intercept 4.30571723486

Process finished with exit code 0
```

Lower the MSE greater the reliability and accuracy. Hence, from the above observations it is clear that a linear regression technique **with intercepts** gives better results and thus is better than one without intercepts.

Problem 3: Experiment with Ridge Regression

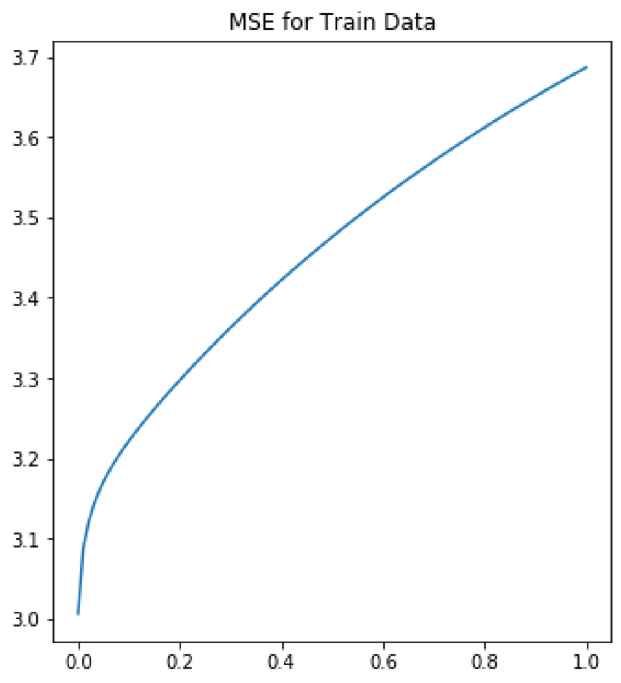
Here are the values for the Training and Testing MSE, while incrementing the value of lambda from 0 to 1 in steps of 0.01 without any regularization. The optimum lambda value is highlighted in red.

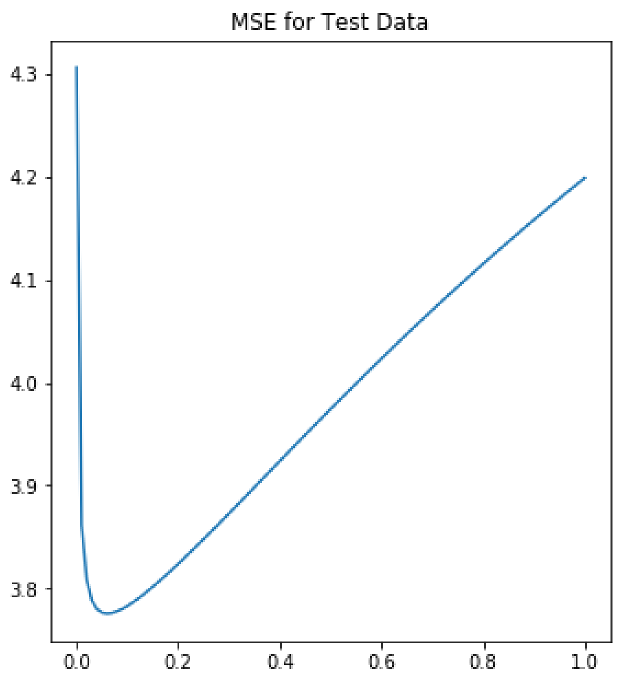
Lambda	TestingMSE	TrainingMSE		Lambda	TestingMSE	TrainingMSE
0	4.305717235	3.006302124		0.5	3.974301219	3.475789994
0.01	3.861635741	3.087452759		0.51	3.979272122	3.480916358
0.02	3.808525691	3.118904862		0.52	3.98422721	3.485997488
0.03	3.78876074	3.140498041		0.53	3.989165896	3.491034009
0.04	3.780211905	3.157124334		0.54	3.99408764	3.496026538
0.05	3.776682231	3.170873668		0.55	3.998991947	3.500975685
0.06	**3.77579807**	**3.182810553**		0.56	4.003878359	3.505882048
0.07	3.776473219	3.193535278		0.57	4.008746458	3.510746217
0.08	3.778147522	3.203410919		0.58	4.01359586	3.515568774
0.09	3.780505536	3.212668787		0.59	4.018426216	3.520350289
0.1	3.783356936	3.221462299		0.6	4.023237206	3.525091328
0.11	3.786580191	3.229896525		0.61	4.028028538	3.529792445
0.12	3.790093853	3.238045337		0.62	4.03279995	3.534454186
0.13	3.793840975	3.245961819		0.63	4.037551203	3.539077089
0.14	3.797780216	3.253684833		0.64	4.042282082	3.543661682
0.15	3.801880528	3.261243291		0.65	4.046992394	3.548208486
0.16	3.806117877	3.268659009		0.66	4.051681969	3.552718014
0.17	3.810473151	3.275948662		0.67	4.056350653	3.557190768
0.18	3.814930781	3.283125149		0.68	4.060998312	3.561627245
0.19	3.81947782	3.290198559		0.69	4.06562483	3.566027933
0.2	3.824103308	3.297176869		0.7	4.070230106	3.570393309
0.21	3.828797817	3.304066457		0.71	4.074814053	3.574723846
0.22	3.833553136	3.310872476		0.72	4.079376598	3.579020008
0.23	3.838362036	3.31759914		0.73	4.083917685	3.58328225
0.24	3.843218089	3.324249929		0.74	4.088437264	3.58751102
0.25	3.848115541	3.330827757		0.75	4.092935303	3.591706758
0.26	3.853049206	3.337335092		0.76	4.097411775	3.595869898
0.27	3.858014386	3.343774052		0.77	4.101866667	3.600000865
0.28	3.863006809	3.350146479		0.78	4.106299974	3.604100078
0.29	3.868022571	3.356453994		0.79	4.1107117	3.608167949
0.3	3.8730581	3.362698045		0.8	4.115101858	3.61220488
0.31	3.878110117	3.368879937		0.81	4.119470467	3.616211271
0.32	3.883175604	3.375000866		0.82	4.123817555	3.62018751
0.33	3.888251783	3.381061936		0.83	4.128143156	3.624133983
0.34	3.893336089	3.387064178		0.84	4.13244731	3.628051067

0.35	3.898426155	3.393008562		0.85	4.136730064	3.631939132
0.36	3.903519791	3.39889601		0.86	4.140991469	3.635798544
0.37	3.908614971	3.404727402		0.87	4.145231582	3.639629659
0.38	3.913709821	3.410503585		0.88	4.149450465	3.643432832
0.39	3.918802605	3.416225378		0.89	4.153648184	3.647208407
0.4	3.923891715	3.421893574		0.9	4.15782481	3.650956725
0.41	3.92897566	3.427508947		0.91	4.161980417	3.654678121
0.42	3.934053059	3.43307225		0.92	4.166115082	3.658372924
0.43	3.939122633	3.438584223		0.93	4.170228887	3.662041456
0.44	3.944183197	3.444045589		0.94	4.174321916	3.665684036
0.45	3.949233652	3.449457058		0.95	4.178394257	3.669300975
0.46	3.95427298	3.454819329		0.96	4.182445998	3.672892582
0.47	3.959300241	3.460133087		0.97	4.186477233	3.676459158
0.48	3.964314561	3.465399009		0.98	4.190488055	3.680001001
0.49	3.969315136	3.47061776		0.99	4.194478562	3.683518401
				1	4.198448852	3.687011647

Choosing a lambda value as **0.06** results in the least MSE value i.e. 0.06 is the optimum lambda

Comparison between the Testing MSE and Training MSE for Ridge Regression:





The MSE for OLE train data with intercept is	3.0063021236
The MSE for OLE test data with intercept is	4.30571723486
The MSE for Ridge Regression train data with intercept is	3.182810553
The MSE for Ridge Regression test data with intercept is	3.77579807

As we can see from the above table, the MSE for the test data when we use Ridge Regression is much lower than that of the OLE method. So, we can declare that the Ridge Regression is a better approach for the given task.

The optimal value of λ is the one, which gives the lowest testing MSE value for the given dataset. In our case, it is 0.06.

Problem 4: Using Gradient Descent for Ridge Regression Learning

The following table shows variation in the values of testing and training MSEs with respect to lambda values. The optimum value is the one highlighted in red.

Lambda	TestingMSE	Training MSE	Lambda	TestingMSE	Training MSE
0	3.811074259	3.165859744	0.5	6.824411196	5.768403
0.01	4.606678392	4.024635279	0.51	6.852286128	5.792504
0.02	4.948843115	4.291748779	0.52	6.879976773	5.81648
0.03	5.129440673	4.429707787	0.53	6.907481602	5.840328
0.04	5.245272376	4.516821206	0.54	6.934799285	5.864043
0.05	5.329050718	4.578934299	0.55	6.961928682	5.887624
0.06	5.394870403	4.627155492	0.56	6.988868833	5.911069
0.07	5.449783795	4.667046316	0.57	7.015618771	5.934374
0.08	5.497693734	4.701686279	0.58	7.042178609	5.957539
0.09	5.540922512	4.732906417	0.59	7.068546772	5.980562
0.1	5.580927334	4.761853248	0.6	7.09472336	6.00344
0.11	5.618663358	4.789274887	0.61	7.12070096	6.026167
0.12	5.654773539	4.815672083	0.62	7.146501927	6.04876
0.13	5.689703057	4.841387509	0.63	7.172103546	6.071199
0.14	5.723766684	4.86666068	0.64	7.197513801	6.09349
0.15	5.757188569	4.891658999	0.65	7.222732474	6.115632
0.16	5.790131868	4.916501333	0.66	7.247760383	6.137624
0.17	5.822713416	4.941270041	0.67	7.272598116	6.159466
0.18	5.855019309	4.966023175	0.68	7.297245329	6.181157
0.19	5.78239872	4.920133558	0.69	7.325902105	6.206301
0.2	5.80021743	4.932646155	0.7	7.34597302	6.224088
0.21	5.818627157	4.945756025	0.71	7.366137811	6.241848

0.22	5.837622269	4.959458399	0.72	7.39388209	6.266354
0.23	5.857197027	4.973748343	0.73	7.417656904	6.28735
0.24	5.877345593	4.988620771	0.74	7.441179478	6.308135
0.25	5.898062039	5.004070451	0.75	7.464518109	6.328771
0.26	5.919340353	5.020092011	0.76	7.487457456	6.349064
0.27	5.941174448	5.036679951	0.77	7.510496362	6.369457
0.28	5.963558163	5.053828649	0.78	7.533438876	6.389778
0.29	5.986485279	5.071532369	0.79	7.556050811	6.409816
0.3	6.009949519	5.08978527	0.8	7.578484094	6.429705
0.31	6.033944558	5.108581413	0.81	7.600739728	6.449447
0.32	6.058464026	5.127914774	0.82	7.622819081	6.469042
0.33	6.083501521	5.147779243	0.83	7.644716052	6.488484
0.34	6.109050606	5.168168641	0.84	7.66645394	6.507794
0.35	6.135104824	5.189076719	0.85	7.68801194	6.526952
0.36	6.161657699	5.210497175	0.86	7.70939857	6.545967
0.37	6.18870274	5.232423652	0.87	7.730615853	6.564838
0.38	6.216233451	5.254849753	0.88	7.751664109	6.583567
0.39	6.244243335	5.277769043	0.89	7.772545418	6.602155
0.4	6.272725896	5.301175057	0.9	7.793256959	6.620599
0.41	6.301674648	5.325061308	0.91	7.813809684	6.638908
0.42	6.59503456	5.571528758	0.92	7.834200268	6.65708
0.43	6.624301561	5.596489344	0.93	7.854426716	6.675111
0.44	6.653404855	5.621360036	0.94	7.874492698	6.693006
0.45	6.682340975	5.646135134	0.95	7.894412939	6.710777
0.46	6.711106725	5.670809313	0.96	7.91415513	6.728395
0.47	6.739699174	5.695377606	0.97	7.933745659	6.745883
0.48	6.768115647	5.719835396	0.98	7.953184063	6.76324
0.49	6.796353717	5.744178395	0.99	7.972468707	6.780466
			1	7.991602785	6.797562

Choosing a lambda value as **0** results in the least MSE value i.e. **0** is the optimum lambda

A comparison of the MSEs for training and testing data is shown in Fig. A. It compares the MSEs obtained using Direct or Squared Loss Minimization with those obtained using the scipy libraries minimize function. The optimum value of lambda is found to be 0 with the minimize function while it is at 0.06 with the squared loss minimization.

Comparison between Testing and Training MSE for Direct and Scipy Minimization:

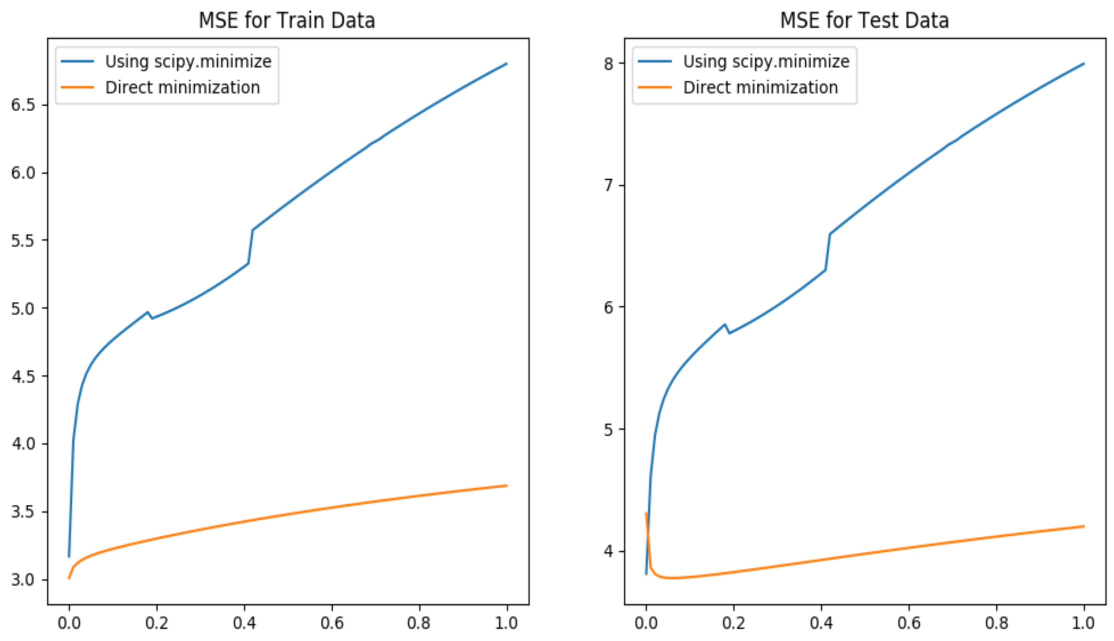


FIG A

Problem 5: Non-Linear Regression

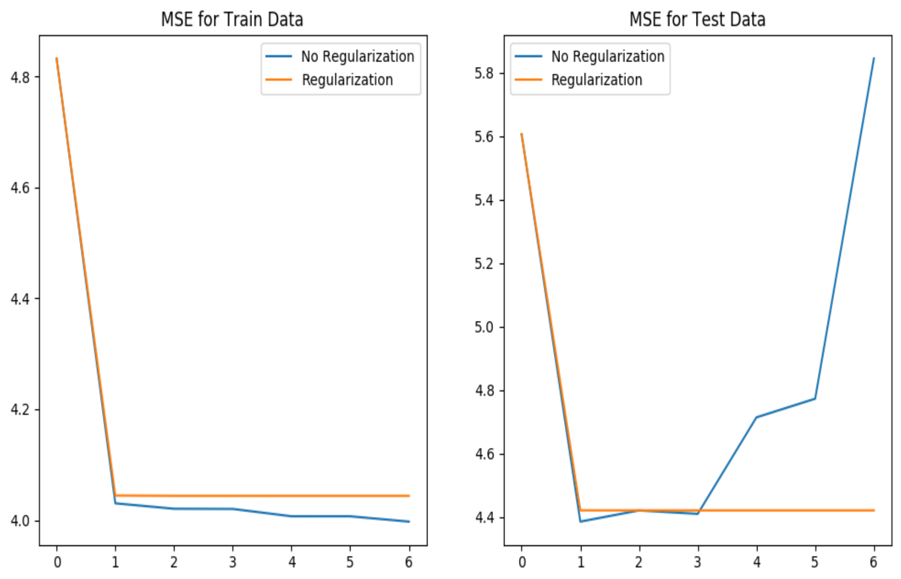
The below table shows the MSEs for training and testing data with and without regularization (Lambda value is set to optimum as obtained in problem 3):

With Regularization				**Without Regularization**			
P	Lambda	TestingMSE	TrainingMSE	P	Lambda	TestingMSE	TrainingMSE
0	0.06	5.60642702	4.832188278	0	0.06	5.6066398	4.83218886
1	0.06	4.38465206	4.030316625	1	0.06	4.413534	4.0410288
2	0.06	4.41991408	4.02052568	2	0.06	4.4133797	4.04043985
3	0.06	4.40906767	4.020191119	3	0.06	4.4133789	4.04043741
4	0.06	4.71345303	4.006953186	4	0.06	4.4133789	4.04043731
5	0.06	4.77222714	4.0069192	5	0.06	4.4133789	4.04043731
6	0.06	5.84527978	3.99735628	6	0.06	4.4133789	4.04043731

As shown in the above table a single attribute is converted into a vector of P values. The optimum lambda value with its corresponding P values are mentioned in the above table.
P in this case is varied from 0 to 6.

In case of Regularization the optimum value of P is 1, while without regularization the optimal P is 3(for lambda=0.06)

The below diagram shows graphs for MSEs of training and testing data with and without regularization with lambda value=0.06(optimum)

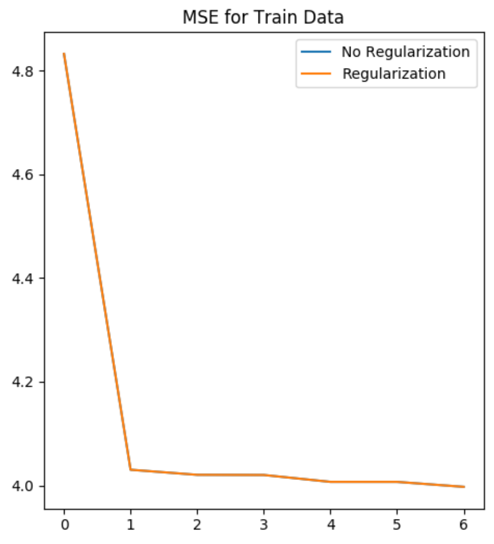


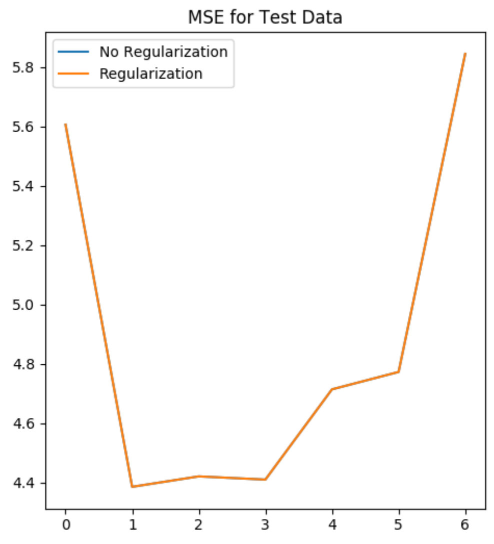
The below table shows the MSEs for training and testing data without Regularization (since lambda is set to 0 only without regularization values and graphs are obtained)

Without Regularization			
P	Lambda	TestingMSE	TrainingMSE
1	0	5.60642702	4.832188278
2	0	4.38465206	4.030316625
3	0	4.41991408	4.02052568
4	0	4.40906767	4.020191119
5	0	4.71345303	4.006953186
6	0	4.77222714	4.0069192
7	0	5.84527978	3.99735628

The optimum value of P is 2 for lambda value as 0

The diagram below shows MSEs for training and test data with lambda set to 0 varying over all the values of P (0 to 6):





From the above mentioned values and graphs it can be observed that the testing MSE for optimum lambda value without regularization is 4.38 while for lambda equal to 0 it is 4.41.

<u>Problem 6: Interpreting Results</u>

The following table compares the Testing MSE and Training MSE values for all the various methods we have used so far in the program:

Technique Used	**Testing MSE**	**Training MSE**
Linear Regression without Intercept	23.10577	8.88388
Linear Regression with Intercept	4.30571	3.006
Ridge Regression (for optimal λ)	3.77579807	3.182810553
Ridge Regression with Gradient Descent (for optimal λ)	3.811074	3.165859
Non-Linear Regression without Regularization	4.4133	4.04043731
Non-Linear Regression with Regularization	4.38465	4.0069192

After comparing all the values, we conclude that Ridge Regression (with intercept) is the best approach among all the other procedures. The Linear Regression performed well in the Training MSE with intercept but when it comes to Testing MSE, it performed bad. If Linear regression is used without an intercept, both the training and testing MSE are worse than almost all the other approaches. Speaking of Non-linear Regression, it performed worse than the Ridge regression.