CSE 574 – Introduction to Machine Learning

Programming Assignment 1

Haripriya Iyer - 50291988

Priya Karadi - 50291880

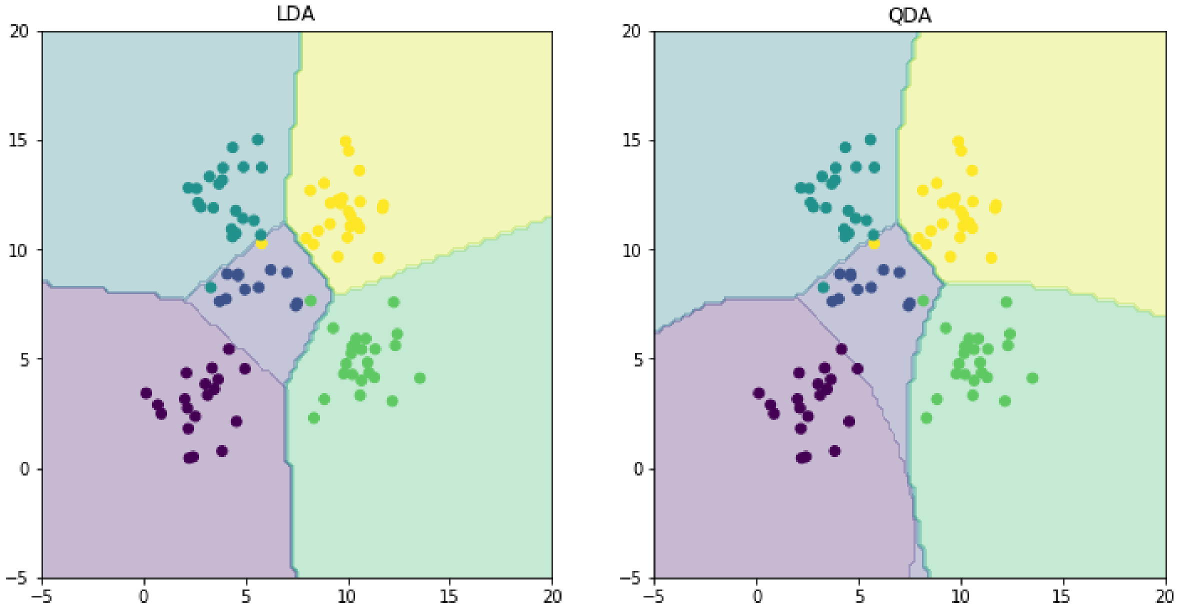
Problem 1: Experiment with Gaussian Discriminators

Gaussian Discriminant Analysis uses a generative learning algorithm where we assume the likelihood of the data points being in a class are distributed as per multivariate Gaussian Normal Distribution. LDA and QDA are used to perform classification of data into multiple classes. For P predictors of the dataset, the mean for each class and variance is calculated to describe the dataset, which is learnt during training.

LDA and QDA vary only in their assumption of the covariance matrix. Essentially, LDA computes only one covariance matrix for the entire set of p-predictors. QDA computes a covariance matrix for each of the k-classes of the p-predictors separately. This leads to a function that is quadratic in X (the input).
QDA suffers from high variance in the data because it calculates variance at individual class level.
The Bernoulli parameters, covariance and means are estimated using the normal distribution formula. Once the parameters are obtained, the MLE is estimated for each class of the multivariate normal distribution. After estimating MLE, we calculate the accuracy, which will give us how well our classifier has performed.

The difference in the boundaries for LDA and QDA is due to the variance between the two methods. LDA does not suffer from high variance in the data, but QDA does.
As the variance is high for QDA since it uses different variance for each class, the equation becomes quadratic in nature and hence the boundary lines become a curve in case of QDA. This is the reason LDA gives a linear boundary and QDA gives a non-linear one.

LDA Accuracy: 97 %
QDA Accuracy: 96 %



Problem 2: Experiment with Linear Regression

A Linear Regression model asserts that the response 'y' is a linear function of input X.
In this case we have a set of training data points X and test data points Xtest, and we build a linear model based on the training data points X, by calculating the weights 'w' or learning the model coefficients of the linear relationship which is done by calculating the maximum likelihood of the training data.

In the first case we directly compute $y = w_1x+w_2x+…+w_{64}x$ or $y = w^Tx$, where w is the weight vector and here we are assuming that the line, y passes through the origin since we have not introduced a bias term.

In the second case we compute $y = w_0 + w^T x$, where we incorporate the intercept or Bias term and here we are not forcing the line to pass through the origin. For this we augment the X matrix with 1, giving [1 x].
If the expected value of y is not 0, then we would also need to estimate the bias weight (i.e., the shift from 0) along with the feature weights. If we don't account for this bias, then any predictions from our estimated model will be off, that is why we calculate the bias term by augmenting the x matrix.

The Means Squared Error for the training and test data of Linear Regression are –

Mean Squared Error (MSE) without intercept : 106775.36155789059
Mean Squared Error (MSE) with intercept : 3707.8401813150163

Here we can see that the MSE for the training and test data is less with using the intercept, and so we can say that MSE with intercept is better. Thus by using the intercept term or the bias term we are ensuring that the predicted data points y, are not passing through the origin. Thus the error is less here.

Problem 3: Experiment with Ridge Regression

In Linear Regression the data points y are predicted by using the Maximum Likelihood Estimation, but this tends to over fit the data. The linear model with increased features tries to estimate wright values w, as close to the training data as possible, which leads to over fitting when we fit the model on testing data. The loss function in the linear model, tend to increase in value as the number of features increases, in order to curtail that we introduce a penalty term λ along with taking an L_2 norm of the weights, which tries to minimize the loss function. This is called regularization.

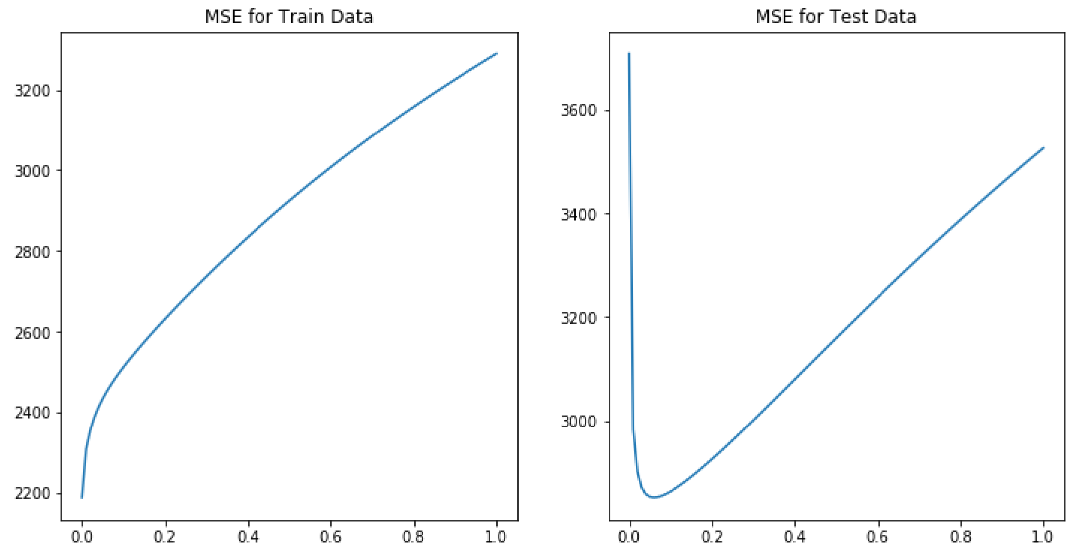
In this experiment we have in the first case plotted training data against λ, which varies from 0 to 1 and as shown in the graph, the MSE value increases with increase in the value of λ for the training dataset.

In the case of Test Data, however, we can see that the MSE value decreases initially for small values of λ, and then increases as λ value increases which denotes that the introduction of the penalty term λ tries to reduce over fitting.

For the Training Data the optimum value of λ is 0, and hence the MSE for the Training Data is the same as the one in case of Linear Regression model which is 2187.16029493.

For the Testing Data, the optimum value of λ is 0.06 where the MSE is the least, which has reduced from 3707.84 as shown in the linear model fit above to a value 2851.33.

Thus the introduction of regularization in Ridge Regression improves the fitting of the test data.



Problem 4: Gradient Descent for Ridge Regression

Gradient Descent is used for reducing the computational cost incurred normally, when we are calculating the inverse of the dataset input - $(X^TX)^{-1}$, when finding out the weights.

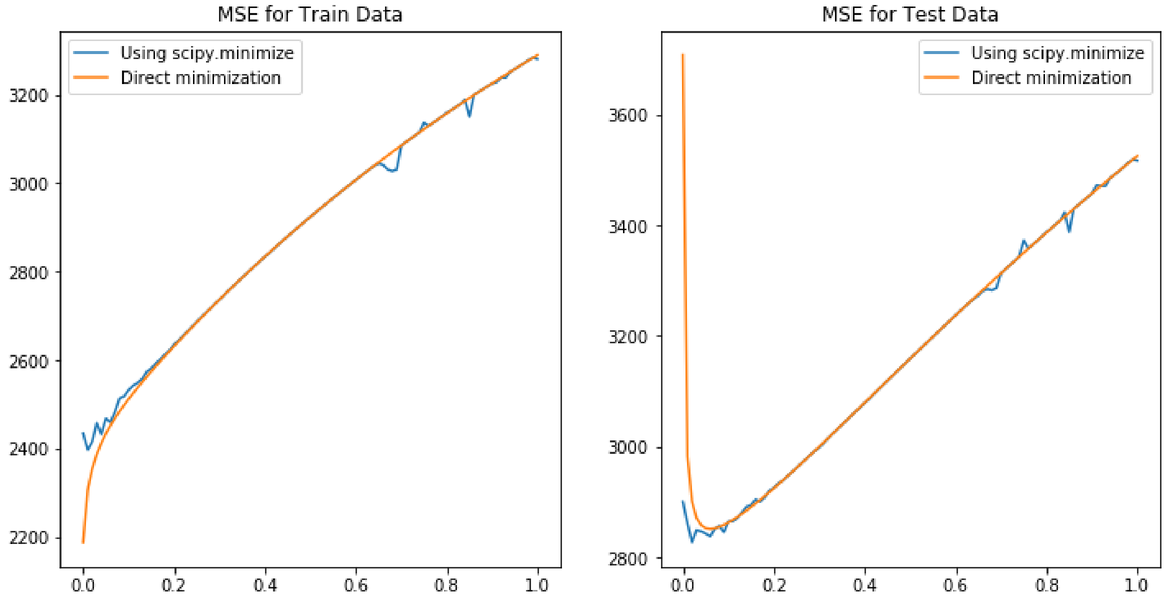
In the function *regressionObjVal*, we first calculate the error or loss function J(w), and then calculate the gradient of it ∂J(w)/∂w, which performs the minimization .

On comparing this graph generated with the problem 3 graphs, it can be seen that the gradient descent performs the same function as calculation of inverse for finding weights since the value of MSE training and test data are varying in the same way as direct minimization performed in the previous problem.

The value of λ=0.01 for Training Data, for which we get the minimum value of MSE of 2396.44210894.
And it can be seen that the value of λ = 0.02 in case of Testing Data is the value for which we get the minimum MSE of 2826.9525654, the result from the minimize function of scipy, which is an optimizing function.

It can also be seen that the MSE has decreased in this case by using Gradient Descent for Ridge Regression.

The dips in the blue curve means that the function is trying to find out the local minimas, and trying to converge at those points, whereas the shoots in the blue curve means that the function is trying to overshoot the MSE for particular values of λ .



Problem 5: Non Linear Regression

A linear Regression model can be converted into a Non Linear Regression model by replacing x with a function of x, so in this case, the weights are linear but the inputs x are non - linear in nature, they can be a polynomial function of degree p. In the given experiment, only the 3rd input variable values are converted into higher order polynomials.

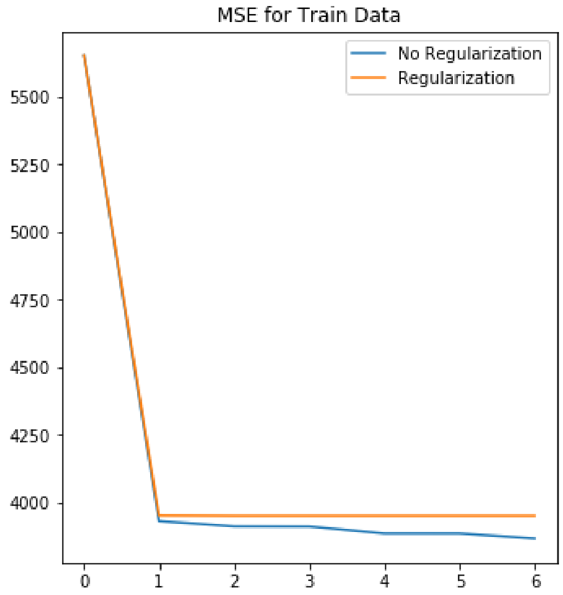
In case one the data is plotted for λ = 0 and here the MSE can be seen to decrease as the value of p increases with and without regularization.

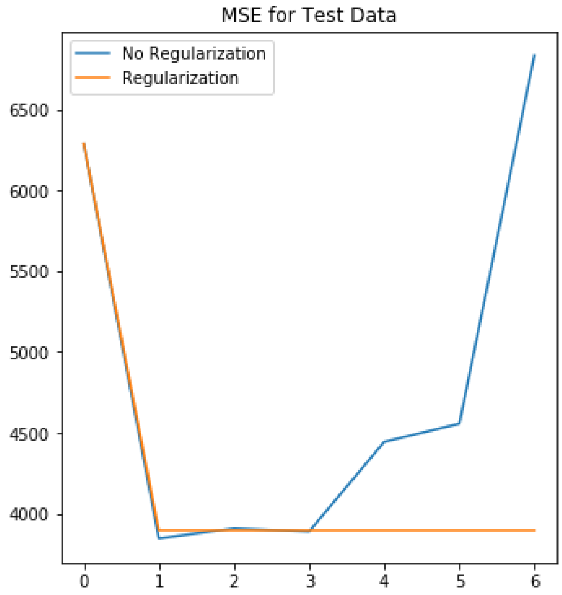
In the second case the data is plotted for the optimum value of λ = 0.06, and here we can see that as p value increases the error decreases in case of regularized data and in case of Non regularized data, the MSE increases after crossing a certain threshold p value.

This shows that as the degree of the polynomials increase the Non-linear Regression curve for Non Regularized data, tends to overfit the data for the test set, here as the value of p increases from 1 we can see that the Training Data MSE is decreasing and remains constant after 1. But the Test Data MSE is increasing after p value 1, which is linear regression case. Thus this issue is solved by regularizing the weights of the data, which is done by ridge regression model and we get a constant MSE for Test data as the polynomial degree increases.

So in case of training data the value of p for λ = 0 or Non Regularization case is 6 where the MSE is the least and the value of p for λ = 0.06, or for the Regularized case, is 1 for least MSE.

In case of testing data the value of p for λ = 0 or Non Regularization case is 1 where the MSE = 3845.03473017. And the value of p for λ = 0.06, or for the Regularized case, is 1 where the MSE = 3895.85646447.





Problem 6: Interpreting Results

The problems 2 – 5 are using the diabetes dataset. To compare the performance of each model, the Test and Training data Mean Squared Error for all these models from Problems 2 – 5 are computed. They are as follows-

Models/Approaches	Training Data	Testing Data
Linear Regression	2187.160294930391	3707.8401813150163
Ridge Regression	2187.16029493	2851.33021344
Gradient Descent for Ridge Regression	2396.44210894	2826.9525654
Non Linear Regression without Regularization	3866.88344945	3845.03473017
Non Linear Regression with Regularization	3950.68253152	3895.58266828

Conclusion:

Thus the recommendations for anyone, using regression for predicting diabetes level using the input features would be Ridge Regression model, using Gradient Descent. Since this approach has the least Mean Squared Error both for Training as well as Testing Data for the diabetes dataset.