CSE 574: Introduction to Machine Learning

Programming Assignment 1:
Classification and Regression

Instructor: Dr. Mingchen Gao
Date: March 14, 2018

Submitted by,
Muthuvel Palanisamy,
Person # - 50246815

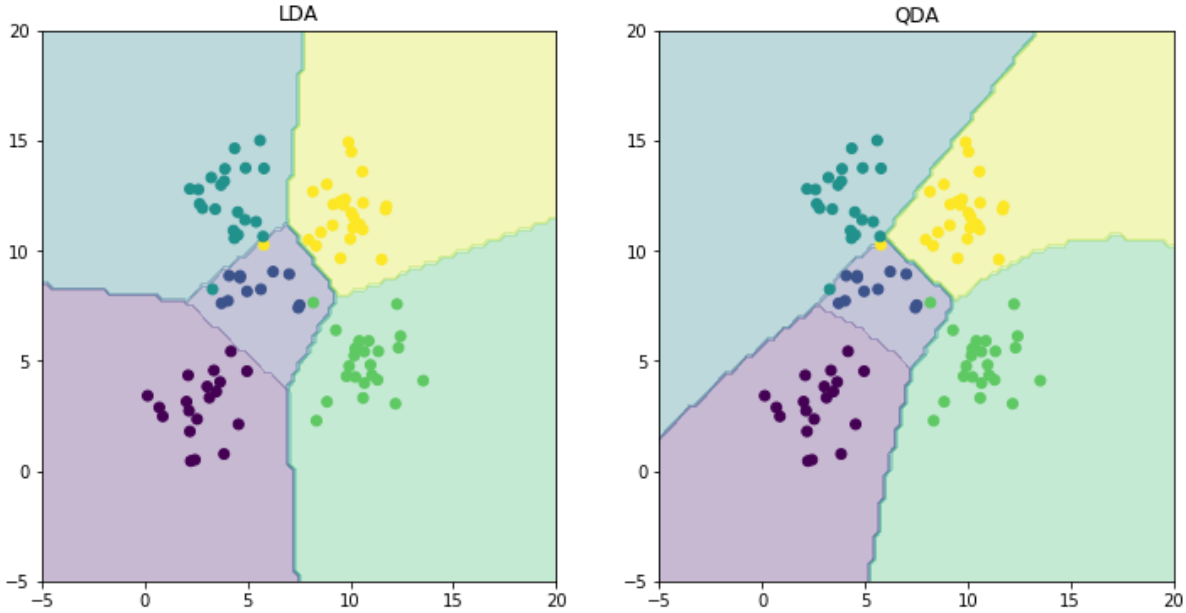
Problem 1: Experiment with Gaussian Discriminators (10 code + 10 report = 20 points)

Implementation**:**

- Used NumPy function (np.mean, np.covariance) to calculate mean and covariance matrices in LDA and QDA
- Dataset: sample_train

Results and Analysis

- **Accuracy**:
  LDA – 97 %
  QDA – 97 %



Discriminating Boundaries for LDA and QDA

- As we could see in the above picture, contour function is used to plot the discriminating boundaries of LDA and QDA
- The points with different colours correspond to different classes

- QDA is able to more accurately plot the boundaries for the different classes
- This is because QDA takes into account the covariance of each classes. This makes the boundary of a specific class to be dependent on the input values of only the specific class
- While in LDA since a single covariance matrix is used across all the test data for prediction, the boundary become a more generic model between classes
- Since LDA produces linear boundaries, it may not be suitable for all datasets unlike QDA
- In the end, QDA proves to be better in differentiating class boundaries

Problem 2: Experiment with Linear Regression (5 code + 5 report = 10 points)

Implementation:

- Performed direct parameter (w) estimation in the learnOLERegression() function
- testOLERegression computes the mean squared error (MSE) of the estimation using the below formula

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

Results and Analysis

- Mean Square Errors of the implementation:
  MSE with intercept: 3707.84
  MSE without intercept: 106775.36
- It is obvious that an intercept in the dataset [1,x] has comparatively low mean squared error and is better comparatively

Problem 3: Experiment with Ridge Regression (10 code + 10 report = 20 points)

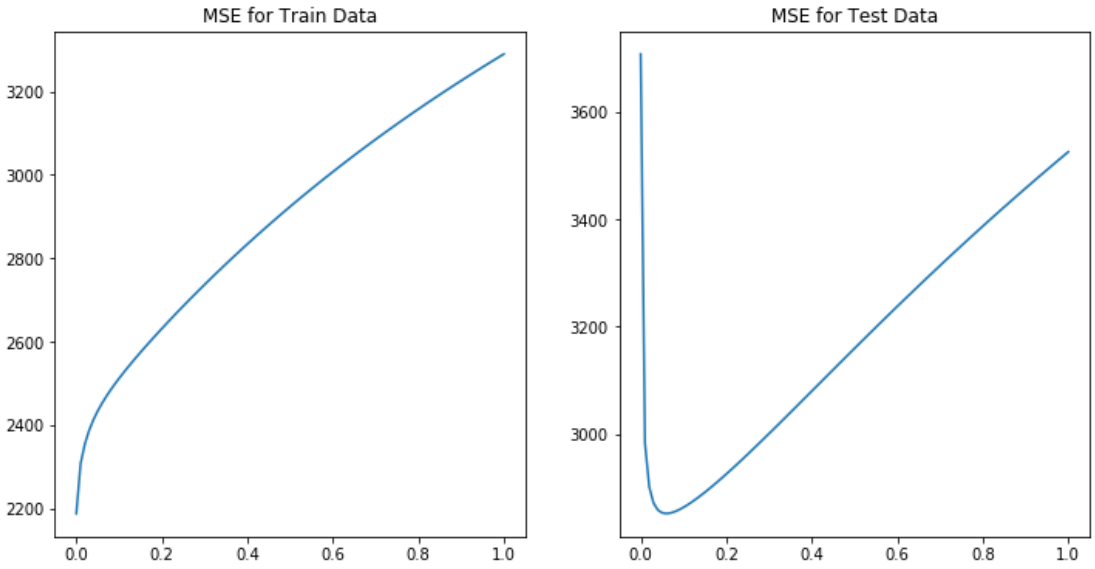
Implementation:

- Implemented parameter estimation for ridge regression by minimizing regularized squared loss

Results and Analysis

- **MSE for train_data** varies from **2150 to 3300** in an increasing manner for **lamba = 0.01 to 0.99** (incremented in steps on 0.01)
- **MSE for test_data** varies from from **2851 to 3700** in an increasing manner for **lamba = 0.01 to 0.99** (incremented in steps on 0.01)
- **MSE for train data** is at it's lowest (=2150) when lambda is 0.00 (No regularization)
- **The optimal value of lambda for train data= 0.00**
- **MSE for test data** is at it's lowest (=2851) when lambda is 0.06
- **The optimal value of lambda for test data= 0.06**
- See the graph below for the plot of Error Vs Lamba for training and testing data

Comparison of MSE vs Lambda for train and test data in Ridge Regression



Comparison of Weight Vectors learnt from Linear Vs Ridge Regression
(Column 0 -> Linear Regression, Column 1 -> Ridge Regression)

	0	1
0	148.155	150.46
1	1.27485	4.80777
2	-293.384	-202.906
3	414.725	421.719
4	272.089	279.451
5	-86639.5	-52.2971
6	75914.5	-128.594
7	32341.6	-167.501
8	221.101	145.741
9	29299.6	496.306
10	125.23	129.948
11	94.4111	88.3044
12	-93.8629	11.2907
13	-33.7283	1.88533
14	3353.2	-2.58364
15	-621.096	-66.8945

(Showing only indices 0 -15 since vector size is big (64))

Comparison of Relative Magnitude of Weight Vectors from Linear vs Ridge Regression

Relative weight - Linear Regression	Relative weight - Ridge Regression
124531.52638433955	959.3129608927522

Linear Vs Ridge Regression (Comparison)

- Lowest MSE for Linear Regression on test data= 3707
- Lowest MSE for Ridge Regression (for optimal lambda = 0.06) on test data= 2851

- MSE for Linear Regression on train data= 3707
- MSE for Ridge Regression (for optimal lambda = 0.06) on train data= 19000
- We could see that introducing the lambda parameter provided a better model with weight vector that has comparability low MSE
- **Ridge regression is better because it has low Mean Square Error**

Problem 4: Using Gradient Descent for Ridge Regression Learning (20 code + 5 report = 25 points)

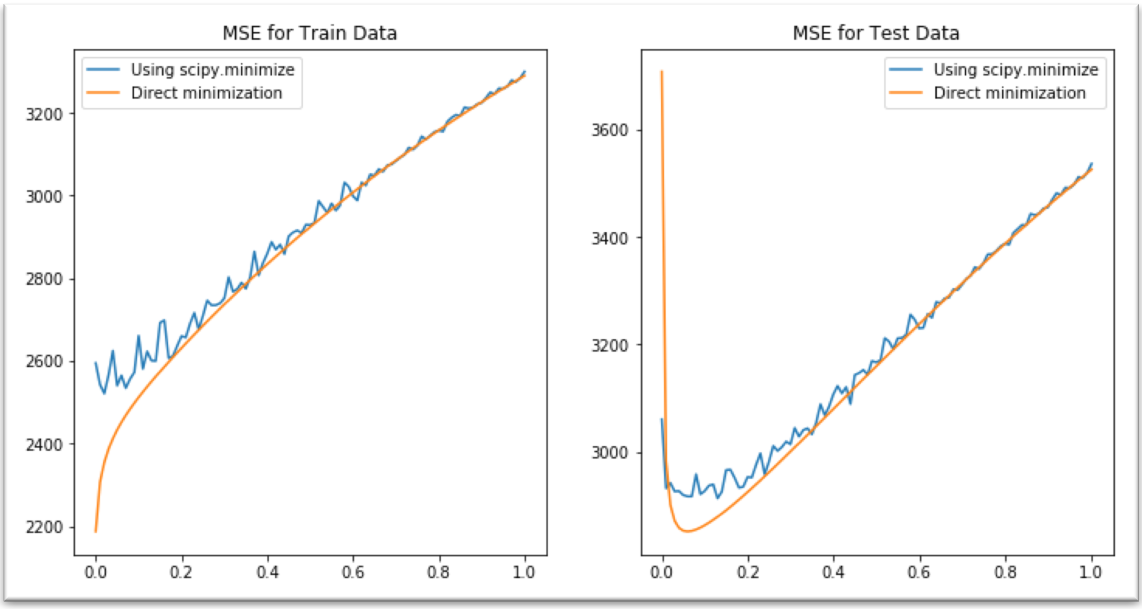
Implementation

- Uses scipy.optimize.minize function to implement gradient descent for ridge regression on the weights

Results and Comparison with Problem 3

- We could see that gradient descent is close to parameter estimation of weights in Problem 3. The gradient reached the optimum value in a few steps and also able to maintain its value without much deviations or fluctuation after a few steps
- Also, gradient descent could help avoid singularity issues in calculating inverse function during direct estimation of weights
- Both the method (gradient descent and direct estimation) reaches optimum after a few steps of lambda

MSE vs lambda for train and test data



Problem 5: Non – Linear Regression (10 code + 5 report = 15 points)

Implementation:

- For every lambda from 0 to 0.99 (incremented in 0.01), non-linear regression in done on data with input values [1, x, x^2, …...]
- P is varied from 0 to 6

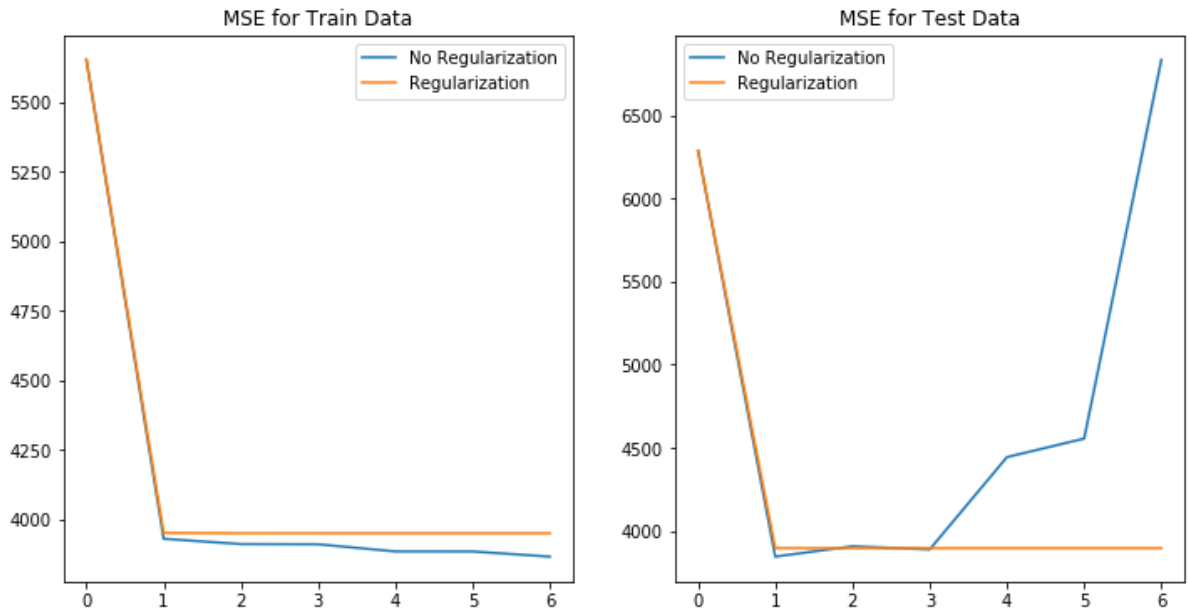
Results and Analysis

Mean Squared Error for test and train data when lambda = 0.06(optimal value)

MSE for test data (lambda = 0)	MSE for test data (lambda = 0.06)	MSE for train data (lambda = 0)	MSE for train data (lambda = 0.06)
6286.4	6286.88	5650.71	5650.71
3845.03	3895.86	3930.92	3951.84
3907.13	3895.58	3911.84	3950.69
3887.98	3895.58	3911.19	3950.68
4443.33	3895.58	3885.47	3950.68
4554.83	3895.58	3885.41	3950.68
6833.46	3895.58	3866.88	3950.68

- **Optimal value of p in terms of test error:**
  - With Regularization – optimal value of p = 1 to 6 (since error is same for p! =0. See graph below)
  - Without Regularization – optimal value of p =1 (error is lowest at this value)
- **Optimal value of p in terms of training error:**
  - With Regularization – optimal value of p = 1 to 6 (since error is same for p! =0. See graph below)
  - Without Regularization – optimal value of p =6 (error is lowest at this value)

MSE vs p for train and test data



Problem 6: Interpreting Results (0 code + 10 report = 10 points)

Recommendation for using Regression for Predicting Diabetes Level:

- Use input with intercept for low MSE (evident from Problem 2)
- Ridge regression is better to address non-linearity of data
- If you are using ridge regression, direct estimation of parameter w is fine unless if the data is correlated which may lead to singularity
- Gradient descent estimation of parameter – w always proves to be effective
- Ridge regression performs well mostly (evident from comparatively low relative magnitude)
- During estimation of parameters using gradient descent, use lower order value for lambda for both training and testing for low error (evident from problem 4)
- Higher order value of p is better during testing and lower order value of p is suitable, for testing non-linear data without regularization (evident from problem6)
- Any magnitude of p! =0 is suitable for Estimation of non – linear data with regularization (evident from problem6)