**ML Assignment-1**

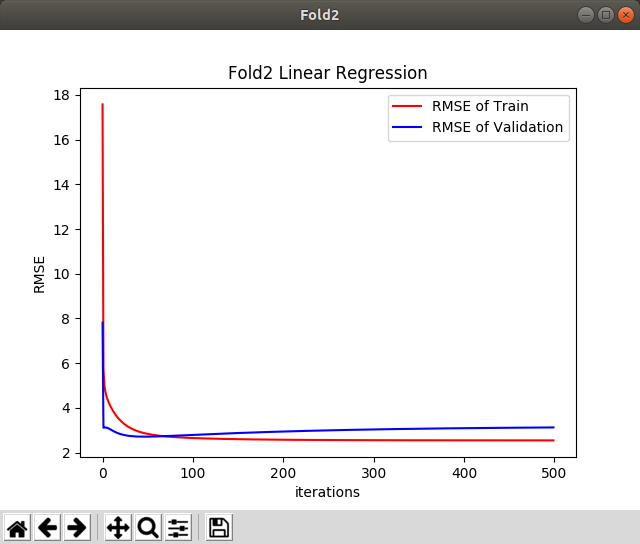
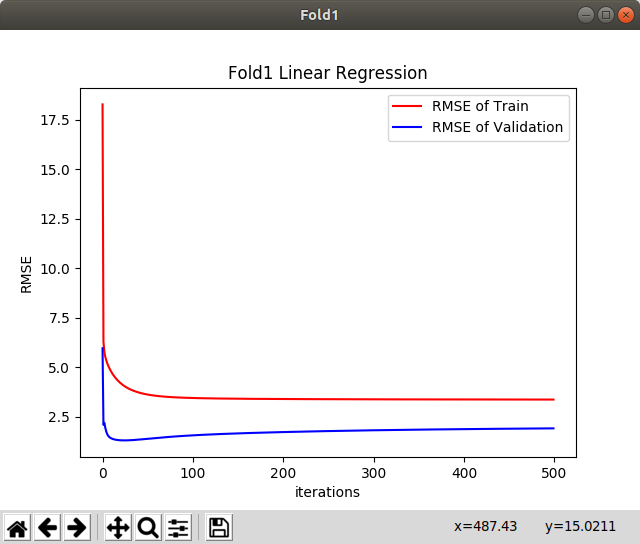
**Observations and Report**

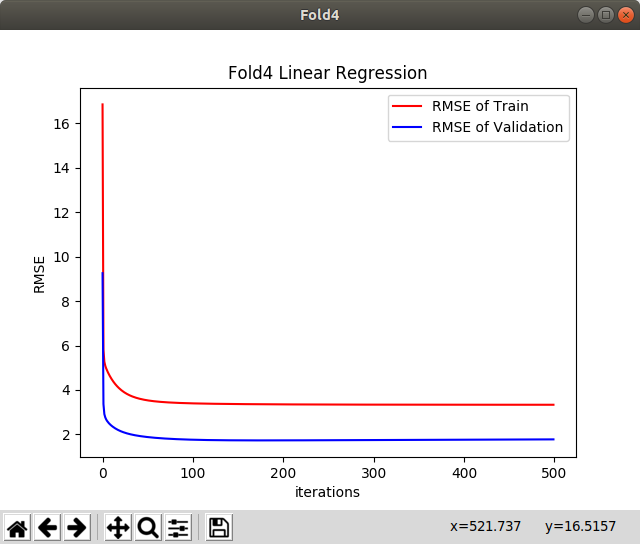
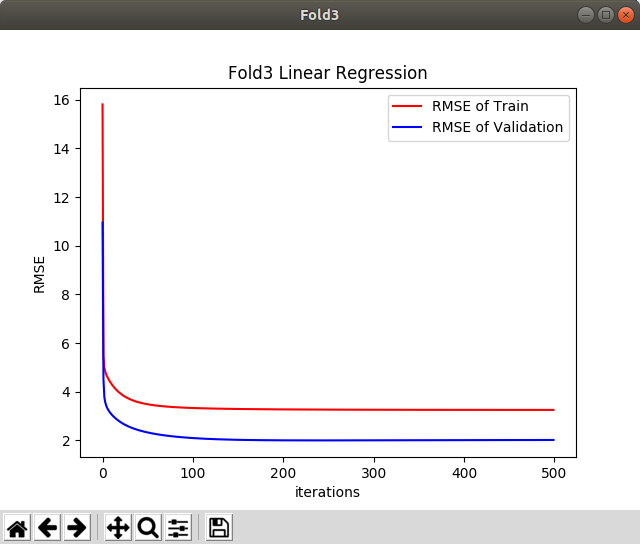
**1.**

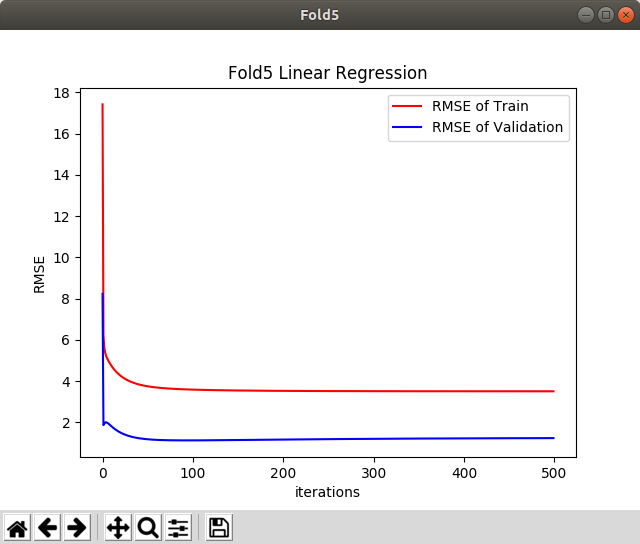
**I. Linear Regression Gradient Descent:**

* Initially load the data from dataset like features(Input examples) and output(actual values- column MV).
* Initialised a theta(coefficient) vector to zeroes and features into X.
* Calculated theta values for each iteration on each 5-folds on using Gradient Descent.
* Calculated RMSE values for each iteration on each 5-folds on both training and validation set and stored in list and ploted graph.
* Observed that for each iteration RMSE values are decreasing i.e., Gradient descent is converging.
* Plotted graphs for each fold, RMSE and Standard deviation.
* Now applied Regularization on that set for both L1(Lasso) Regression and L2(Ridge) Regression and plotted the graphs for both.

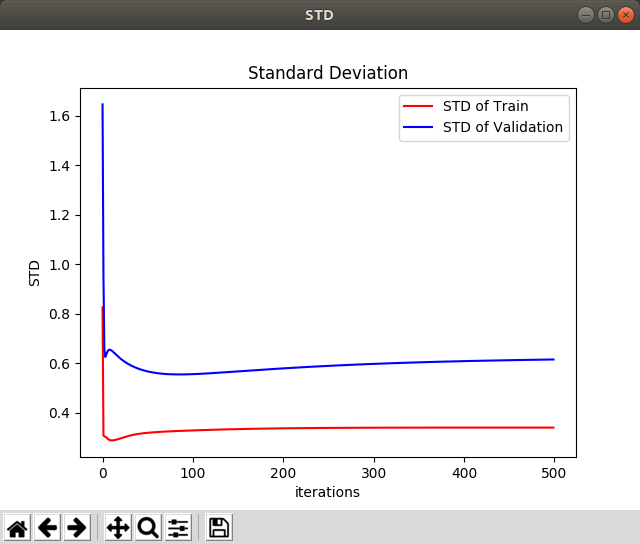
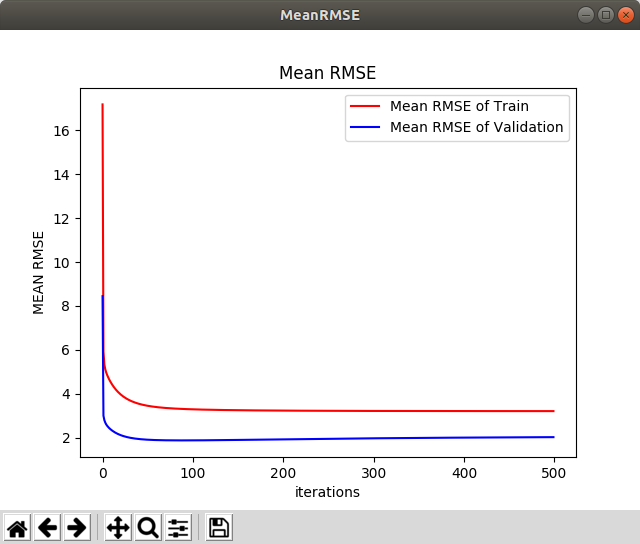
The RMSE values plotted for each training set over each iteration until the error has converged:



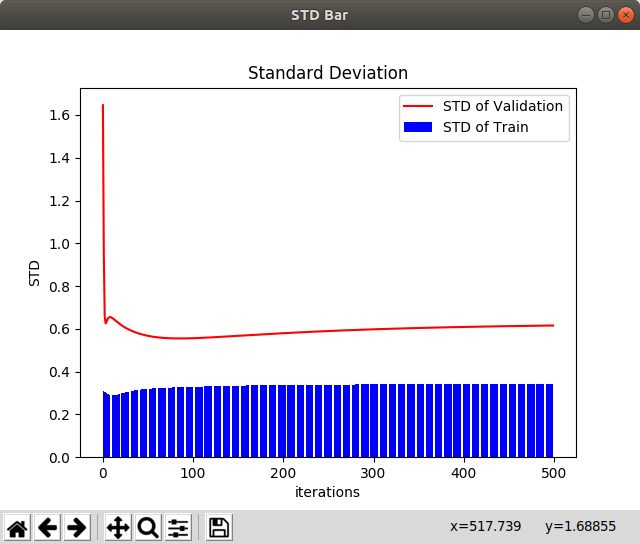




Mean RMSE and Standard deviation:

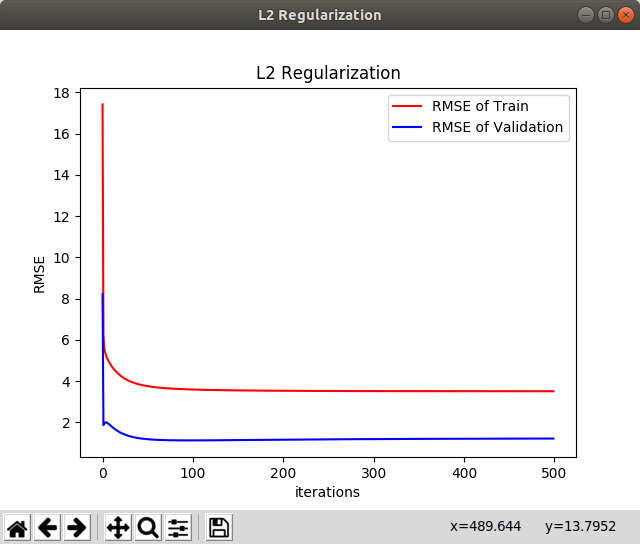
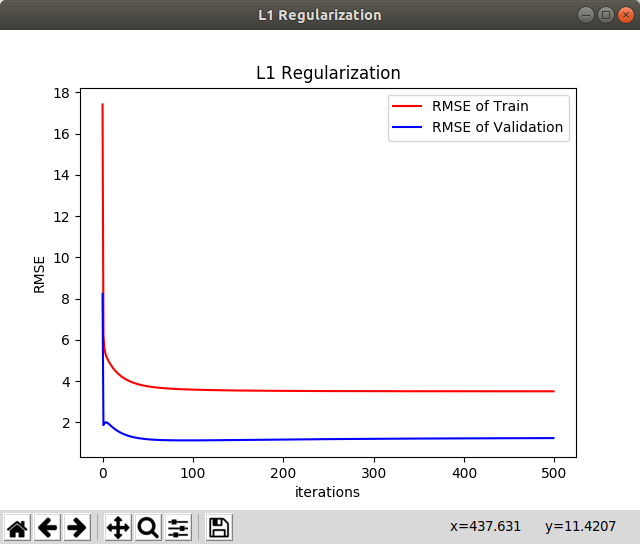


Graph using Bar



**II**. Regularization:

* The each 5- folds have been trained and by taking the best RMSE valued fold and trained the model using Hyperparameter(Lambda) by using L1 and L2 regularization and plotted the graphs for each regularization.



**III**.

No model is Overfitting and Underfitting but by using Regularization the error value is slight better than without Regularization model.

**2.**

**I.**

Logistic Regression using L2 Regularization:

* Implemented using Logistic regression in SKLEARN by solver=’lbfgs’
* Here I implemented it using one vs rest like taking one class as ‘1’ and others as ‘0’ and giving trainImages and trainLabels its accuracy is reduced but accuracy of Test is high.

Class: 0

TrainScore: 0.0037166666666666667

Test score: 0.9922

Class: 1

TrainScore: 0.20803333333333332

Test score: 0.9933

Class: 2

TrainScore: 0.09891666666666667

Test score: 0.9802

Class: 3

TrainScore: 0.09878333333333333

Test score: 0.9761

Class: 4

TrainScore: 0.0987

Test score: 0.9834

Class: 5

TrainScore: 0.09816666666666667

Test score: 0.9779

Class: 6

TrainScore: 0.09778333333333333

Test score: 0.9846

Class: 7

TrainScore: 0.09875

Test score: 0.9833

Class: 8

TrainScore: 0.09928333333333333

Test score: 0.9464

Class: 9

TrainScore: 0.0988

Test score: 0.963

**II.**

Logistic Regression using L1 Regularization:

* Implemented using Logistic regression in SKLEARN by solver=’liblinear’
* Here I implemented it using one vs rest like taking one class as ‘1’ and others as ‘0’ and giving trainImages and trainLabels its accuracy is reduced but accuracy of Test is high.

Class: 0

Train Score: 0.0032

Test Score: 0.9912

Class: 1

Train Score: 0.20831666666666668

Test Score: 0.9922

Class: 2

Train Score: 0.09878333333333333

Test Score: 0.9794

Class: 3

Train Score: 0.09871666666666666

Test Score: 0.9778

Class: 4

Train Score: 0.09873333333333334

Test Score: 0.9834

Class: 5

Train Score: 0.09811666666666667

Test Score: 0.9772

Class: 6

Train Score: 0.09788333333333334

Test Score: 0.9844

Class: 7

Train Score: 0.09871666666666666

Test Score: 0.9825

Class: 8

Train Score: 0.0998

Test Score: 0.961

Class: 9

Train Score: 0.09878333333333333

Test Score: 0.9672

**III.** For TestSet -the models neither overfit nor underfit since regularisation prevents such anomalies by introducing a hyper-parameter to regularise our prediction model.

But for taking One vs Rest method the accuracy on TrainSet is less like Underfitting.