Linear Regression Observations

Here we have implemented Linear Regression to predict the age of snail Abalone.

DataSet Creation: - We read the given dataset into a matrix and randomly shuffled the rows of this matrix then we partitioned the data into training and testing sets using fraction value 0.3 initially. This means that 30% of the given data will become training data and the rest 70% will be testing data.

Data standardization: For standardizing the data we have used the below formula

For all i from 1 to N: $(x_i$ –mean(x))/standard deviation(x)

After partitioning the data and standardizing it we have added one more column, with all values equal to 1, in the beginning of the dataset to form X_o=1 for W_o.

We have tried getting the weights for different sets of lambda. All those sets are given below:

```
Lambda = [0 0.1 0.001 0.00001 0.0000001 0.000000001];
Lambda = [0 0.1 1 2 5 10 20];
Lambda = [0.001 0.01 0.1 0.5 1 5 100 500 1000]; %%%% as we increase lambda
weights tend to zero
Lambda = [0.1 0.3 0.5 0.7 0.9 1.1 1.3];
Lambda = [0.001 0.1 1 2 5 10 20];
```

Important Observations:-

Action	Impact
Increase Lambda	Most of the weight coefficients approach to 0
Impact of increase in lambda on mean error	Slop of the line decreases
Impact of increase in training fraction on mean error	Lines for different lambda tent to converge.
Impact of increase in training fraction on minimum mean testing error	Minimum mean testing error
Slop of line between actual and predicted labels	45°

Result of increasing Lambda: First row denotes the different values of Lambda and subsequent values in each column denotes values of weights resulting from those Lambda.

	1	2	3	4	5	6	7	8	9
1	1.0000e-03	0.0100	0.1000	0.5000	1	5	100	500	1000
2	9.8740	9.8739	9.8732	9.8701	9.8661	9.8348	9.1448	7.0593	5.4933
3	-0.0129	-0.0128	-0.0118	-0.0074	-0.0020	0.0345	0.1976	0.2382	0.2306
4	0.7968	0.7967	0.7952	0.7886	0.7808	0.7283	0.4700	0.3360	0.2864
5	0.9549	0.9550	0.9555	0.9579	0.9607	0.9763	0.9052	0.5674	0.4251
6	3.0198	3.0169	2.9875	2.8634	2.7218	1.9470	0.2732	0.1725	0.1799
7	-3.9669	-3.9655	-3.9513	-3.8912	-3.8217	-3.4153	-1.5733	-0.4668	-0.1806
8	-0.7119	-0.7113	-0.7047	-0.6770	-0.6456	-0.4790	-0.1556	0.0315	0.0997
9	1.6094	1.6103	1.6196	1.6583	1.7018	1.9200	1.5845	0.7361	0.5023
10	-0.2154	-0.2154	-0.2156	-0.2165	-0.2175	-0.2237	-0.2629	-0.2663	-0.2430
11	0.0783	0.0784	0.0786	0.0798	0.0812	0.0901	0.1542	0.1853	0.1729
12	0.1344	0.1344	0.1343	0.1340	0.1336	0.1310	0.1069	0.0799	0.0692

As we can see from the above table that as we increase the value of Lambda most of the weights approach to zero.

$W=\text{inverse}\ (\ X^T * X\ +\ \lambda * I\ )\ *\ X^T\ *\ Y\ ;$

With the increase in lambda the flexibility of the model will decrease which leads to decreased variance but increased bias.

The dependency on each attribute decreases as we increase the value of lambda.

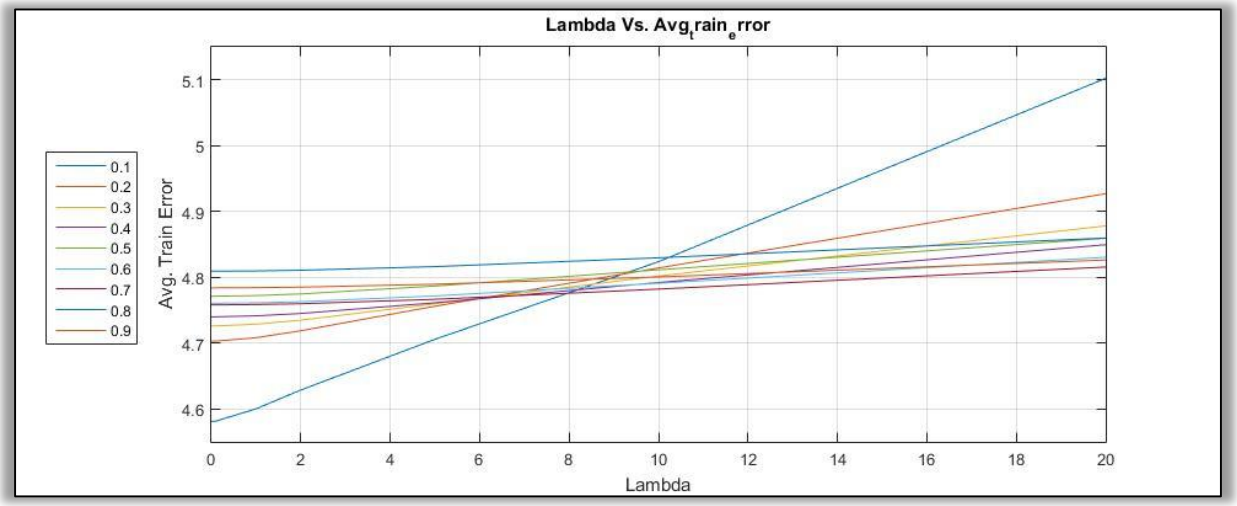
Effect of removing least significant attributes from the dataset:

We are randomly selecting a lambda value and then for the calculated weights we are removing those attributes for which weight is less.

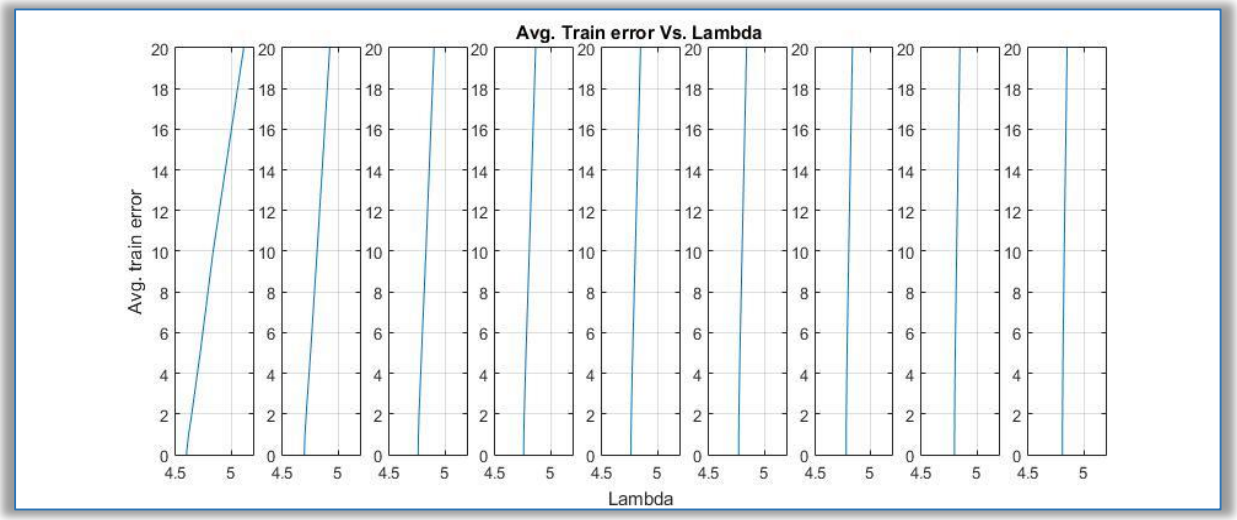
When we remove 3 of the least significant attributes from the dataset the error tend to increase lightly both for the test as well as training dataset. This is happening because these least significant attributes are also playing role in the error calculation, as we have lambda value not very high thus weights are large and thus every attribute have significance.

Graph showing relation between Lambda and Mean Training error:

A.) Each line represents relation between lambda and avg. train error for different values of fraction used to divide dataset between train and test data.



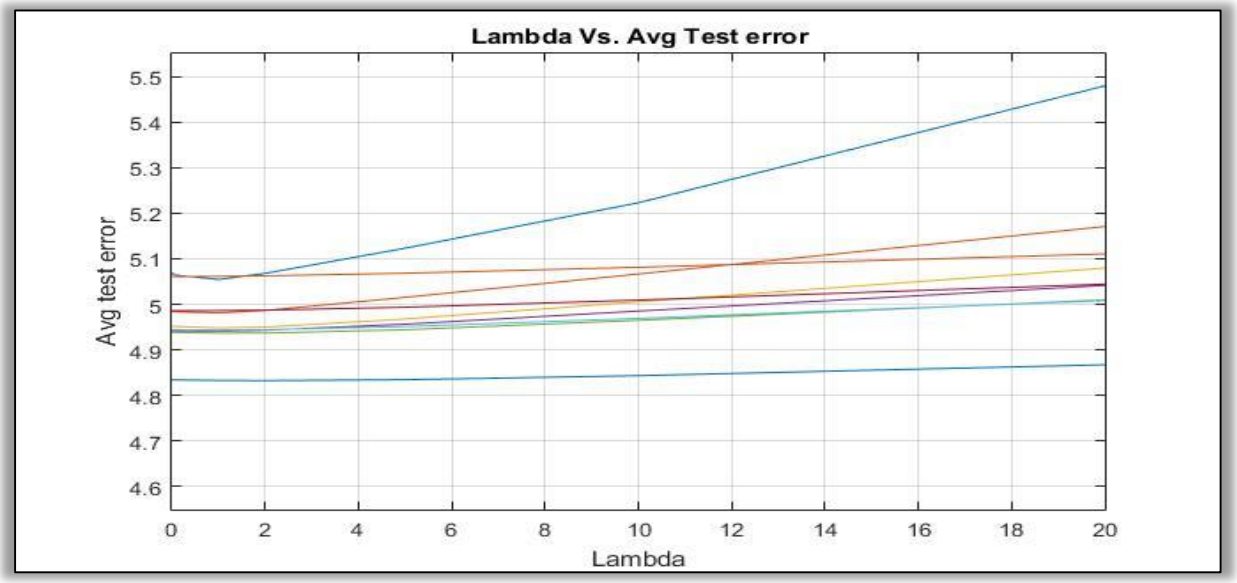
B.) Each sub-graph shows the relation between the Lambda and Avg. Training error for different values of fraction.



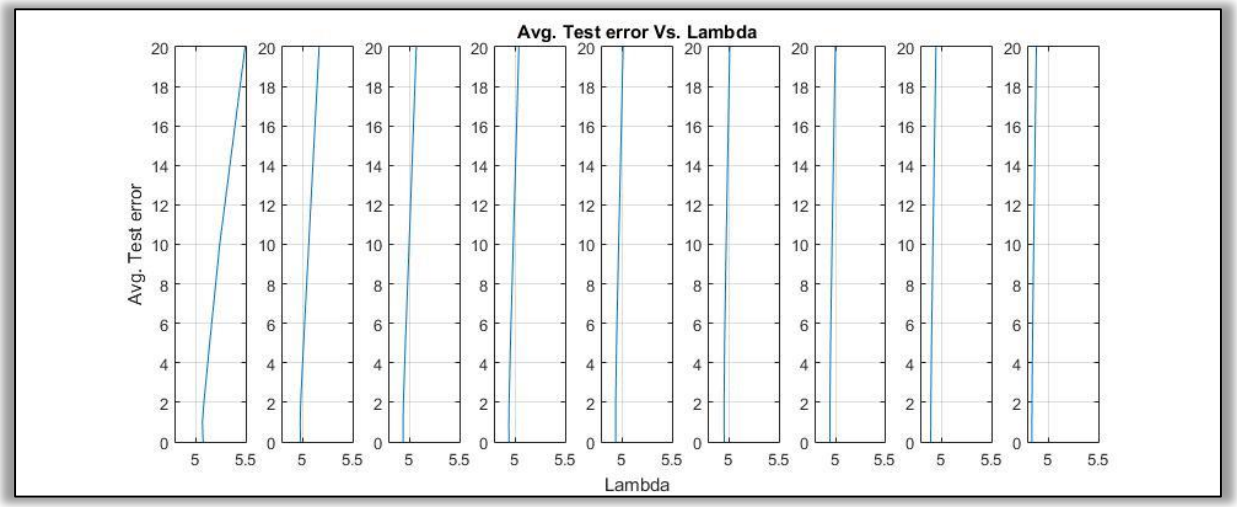
From the above graphs we can see that as we increase the proportion of training data slope of the relation between Lambda and Avg. training error increases.

Graph showing relation between Lambda and Mean Testing error:

A.) Each line represents relation between lambda and avg. test error for different values of fraction used to divide dataset between train and test data.

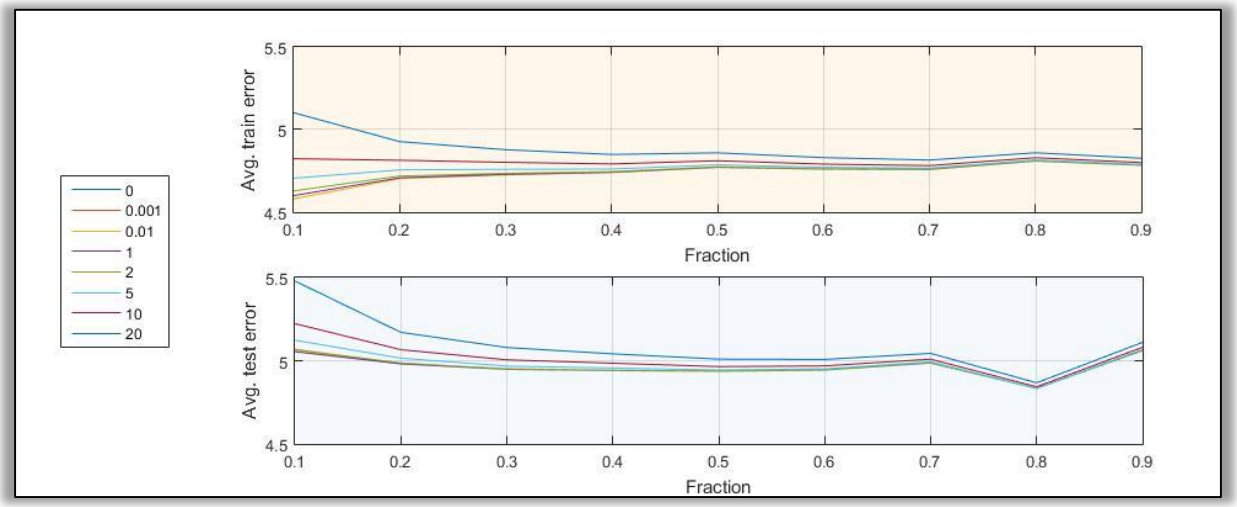


B.) **Each sub-graph shows the relation between the Lambda and Avg. Testing error for different values of fraction.**



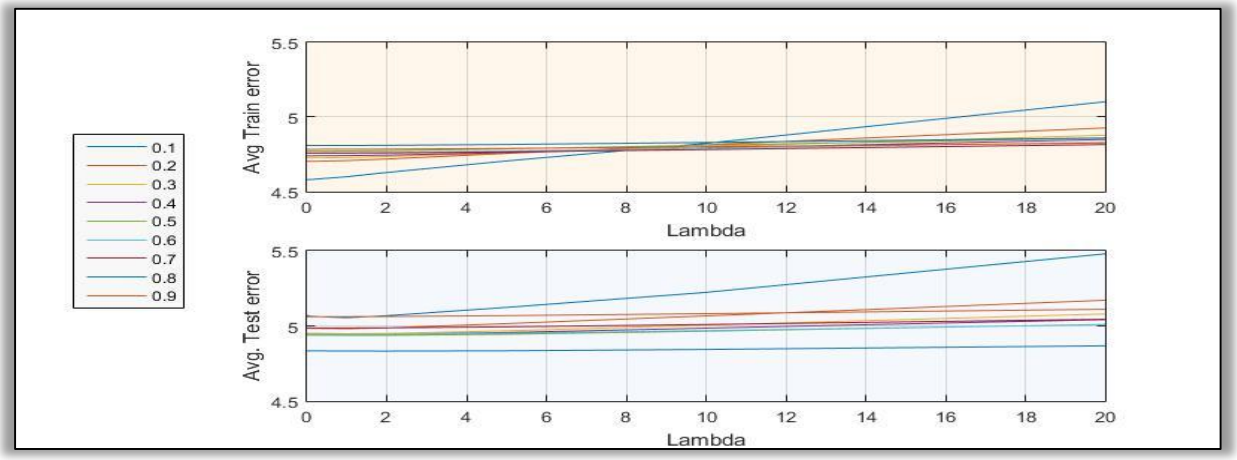
From the above graphs we can see that as we increase the proportion of training data slope of the relation between Lambda and Avg. training error increases.

Graph showing relation between Fraction value to divide dataset into test and train data verses average training error and average testing error respectively :-



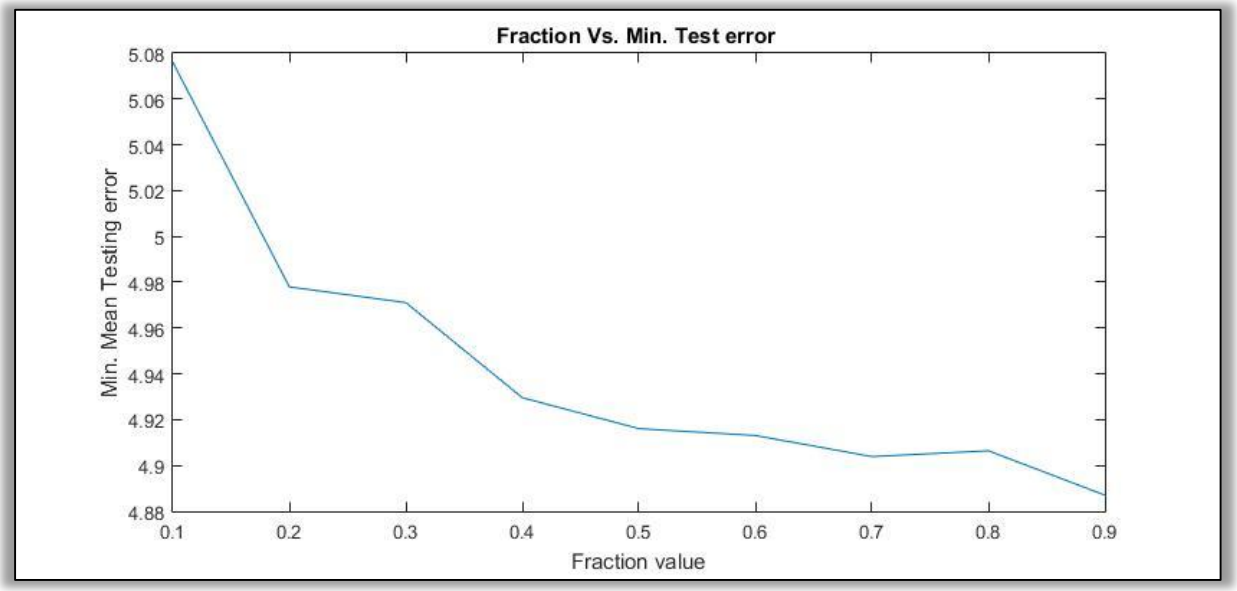
From the above graph we can see that the lines drawn for different lambda which shows the relation between Avg. training/ testing error converges as we increase the proportion of training data. This graph is drawn using same ylim.

Graph showing relation between Lambda verses average training error and average testing error respectively:-



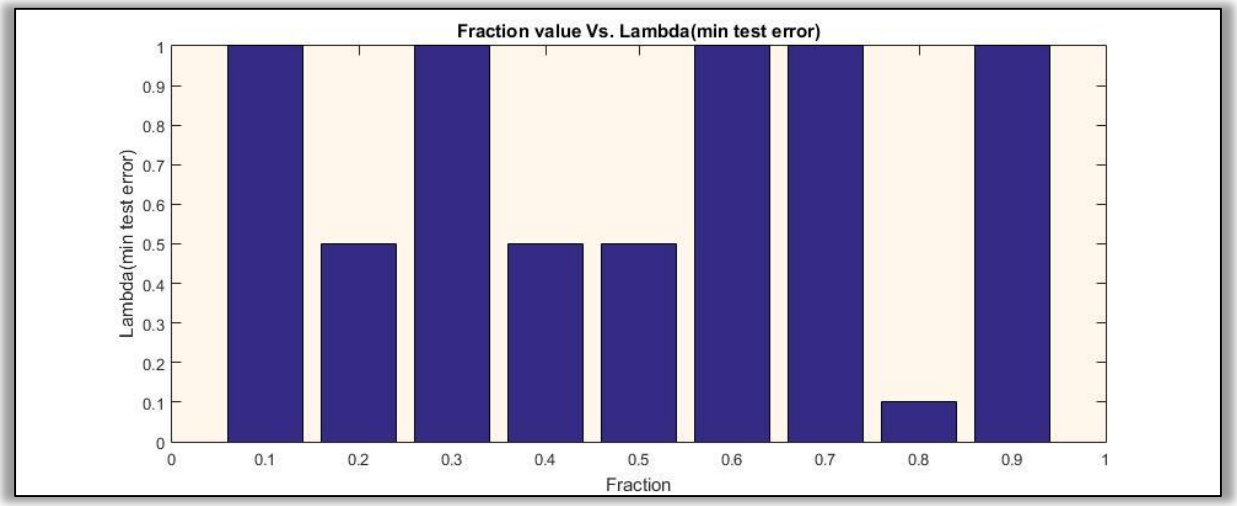
The above figure shows the relation followed by lambda and avg. training/testing error using same ylim.

Graph showing relation between fractions used to divide dataset into train and test data verses corresponding minimum value of mean testing error:



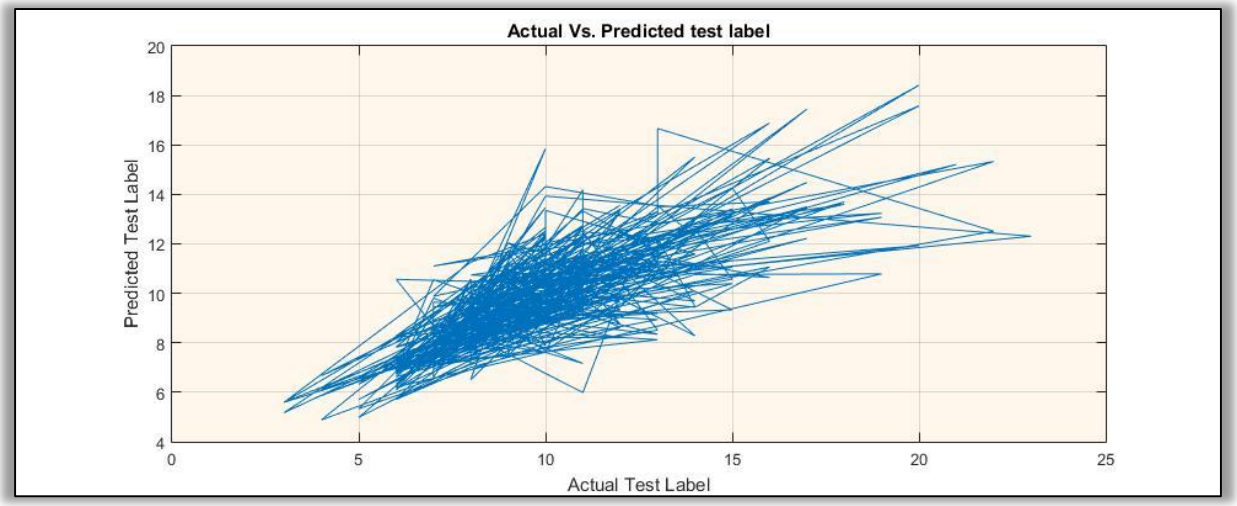
The value of minimum mean testing error decreases as we increase the proportion of training data. This happens because more the data we use for training implies better learning of the model.

Graph showing relation between fractions used to divide dataset into train and test data verses Lambda values that produce minimum mean testing error

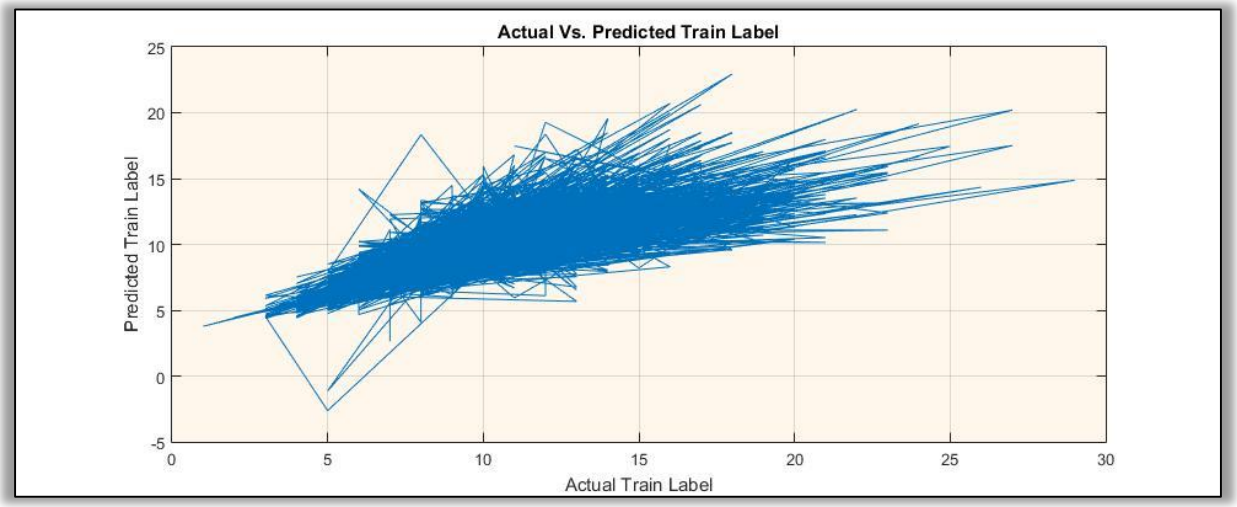


The above figure shows that in most of the cases we are getting the minimum mean testing error at Lambda=1.

Graph showing relation between the Actual and the target Test Labels:-



Graph showing relation between the Actual and the target Train Labels:-



As almost all the lines in the above figures are at 45^{o} slop, therefore we can say that we have learnt a good model. This model is predicting the target labels almost same as the actual target labels for both training and testing data. These graphs are drawn for the fraction value for training data as 0.9 and Lambda=1.